POLITECNICO DI TORINO

Dipartimento di Elettronica e Telecomunicazioni

Master's degree
in
ICT For Smart Societies

Master Thesis

Data driven evaluation of quality of service in mobile radio networks via machine learning

Supervisor
Prof. Marco Mellia
Ing. Maurizio Graziano

Candidate
Matteo Novembre
242273

April 2019

Accademic Year 2017-2018

Summary

Summary 2
1 Introduction 4
1.1 Scenario and Motivation 4
1.2 The Approach 5
1.3 Machine Learning 6
1.4 The Structure 7
2 CSFB Overview 9
2.1 Introduction 9
2.2 LTE Growth and Challenges 9
2.3 Voice communication services evolution 12
2.4 How does the CSFB work? 13
2.5 CSFB Challenges 15
3 Data Preparation - Correlator & Comparator 19
3.1 Introduction 19
3.2 Benchmarking tests 20
3.2 Data extraction 22
2.4 Data mining – Correlator 24
3.5 Data mining – Comparator 30
3.6 Trend Results 33
4 Anomaly detection 41
4.1 Introduction 41
4.2 What are anomalies? 41
4.3 Challenges 43
4.4 Type of Anomaly 44
4.5 Anomaly Detection Technique 46
4.6 Output of Anomaly Detection 49
5 Application and results 50
5.1 Introduction 50
5.2 Problem definition and dataset extrapolation 50
5.3 Related Work 55
5.4 Classifiers comparison 56
5.4.1 Elliptic Envelope 56
5.4.2 Isolation Forest 57
5.4.3 Local Outlier Factor (LOF) 57

5.4.4 One Class-SVM 58
5.4.5 K-Means 60
5.5 Experimental Evaluation 62
5.5.1 Data Selection 62
5.5.2 Pre-processing 65
5.5.3 Application 65
5.6 Results and Performances 68
5.7 Anomalies Localization and Trend Visualization 80
6 Conclusions 85

1 INTRODUCTION

1.1 SCENARIO AND MOTIVATION

In a world in continuous evolution from the telecommunications point of view, increasingly oriented to the development of technologies and infrastructures that allow an improvement of the lifestyle of human from all points of view (just think of the advent of 5G and IoT - Internet of Things), mobile operators are constantly looking for new services and possible optimization of those already provided. Among these services, the data service stands out without any doubt, since the state-of-the-art is increasingly oriented towards the complete use of this in all possible contexts, including the messaging service and the voice service.

Currently the most advanced technology on the market is LTE. LTE stands for Long Term Evolution. This is the currently most advanced mobile wireless communication standard on the market (4G), ensuring higher data transfer rates than previous standards, increased channel capacity and lower latency. One of the main innovations coming with LTE is the so-called VoLTE.

Through the Voice-over LTE, or VoLTE, is possible to make voice calls directly on the LTE network, thanks to the use of an infrastructure full IP Multimedia Subsystem (IMS) where users can place phone call over the LTE network as data packet.

Unfortunately, nowadays this kind of solution is not completely available, forcing in this way the adoption of intermediate voice solutions like *Circuit Switched FallBack* (CSFB). CSFB is a technology whereby voice and messaging services are delivered to LTE devices using GSM or UMTS network. This is necessary because LTE is a packet-based full-IP network that cannot support circuit-switched calls. So, when an LTE device is used to make or receive a voice call or SMS, the device switch (*fall-back*) from LTE network to 3G/2G legacy network to complete the service.

Mobile operators are interested in the analysis of benchmark data relative to the quality of the voice and messaging service operating through the CSFB technology, this being a temporary solution subject to possible errors, mainly during the paging phase or during the catching operation to the cell.

Generally, mobile operators delegate these benchmarking operations to consulting companies, which are equipped with all the necessary equipment and staff of analysts, who are experts in the analysis of this type of data. One of these consulting companies is **Reply S.p.a**., a company specialized in the design and implementation of solutions based on new communication channels and digital media, where this work thesis entitled ‘Data driven evaluation of quality of service in mobile radio networks via machine learning’ was carried out.

Benchmark operations are done on the Italian territory and aim to verify the quality of the voice service provided by various mobile operators. These data are collected using specific equipment, which involves the use of cars equipped with mobile benchmarking fixture. These tools are supplied by **Rohde & Schwarz SwissQual AG**, a telecommunications company that provides and develops tools for benchmark tests.

During the year several benchmark missions are carried out in a given area. In this way, mobile operators can fix or optimize some parts of the infrastructure present in that area. However, these missions generate a huge amount of data. Currently data is analyzed by analysts using specific software, which allow a deep vision of the phenomenon (modulation, interference, etc.). The cases that are analyzed are only bankruptcy events (failed or dropped phone calls). Therefore, the task of analysts is to find the root cause of these failed events. This operation, despite the use of specific software, is particularly laborious and above all cannot be done in a short time.

For this reason, companies like Reply S.p.a. are interested in developing algorithms in order to optimize and automate these laborious operations of cataloguing, providing faster feedbacks to the clients, in this case mobile operators, about the status of their infrastructures.

1.2 The Approach

To do that, it was decided to integrate some algorithms into them, based on specific logics dictated above all by knowledge and experience analysts, allow a lightening but at the same time an optimization of the work of the latter. These logics are defined in such a way that it is possible to identify, with the greatest possible accuracy, the root cause of the faults occurred, each of which is recognized with a unique identifier.

Once this *cataloguing* operation has been carried out, results are compared with those ones obtained by the analysts for the same faults, and an accuracy index is therefore computed. Having reached a good level of accuracy (percentage of the algorithm's output matched to that of the analysts), around 80%, we start to use machine learning algorithms.

1.3 MACHINE LEARNING

Never like in these years it has been possible to see a growing interest from the most different institutions (universities, research laboratories, companies in the IT sector and not only) in the field of artificial intelligence and **machine learning**. This technological innovation lays the foundation, and not only, for an unprecedented development of the companies from the point of view of technology and automation. The fields that affect these areas are among the most varied, starting from the health sector, home automation (*domotic),* transport and so on.

The term "Artificial Intelligence" (AI) was first coined in the 1956 by John McCarthy [1] and indicates all those skills typical of human intellect that are implemented by a machine, almost as if the machine were able to develop their own thinking and make all decisions independently.

Machine learning is simply a branch of the artificial intelligence. With the term "machine learning" we intend "*the ability of a machine to learn without being explicitly programmed*" (Arthur Samuel, 1959). Machine learning is therefore a way to "educate" an algorithm so that it can learn from various environmental situations. The training phase of machine learning algorithms is complex and often requires a large amount of data as it is necessary to adapt (and improve) it according to the situations that occur.

Usually, when there is a set of *N* cases, these are divided into two subsets, the first one is used for training the machine, where the machine learns which are the parameters to use in order to classify an item of the dataset belonging to a specific set or to estimate a specific feature, while the second one is used to test the machine and verify how well it learned by the first subset. In this way we can evaluate the accuracy the testing phase had in estimating the feature of classifying or estimate the item.

Different approaches of machine learning can be found in the literature, but in the case of the thesis work, we decided to use an **anomaly detection** approach. Specifically, anomaly, or outlier, detection is a technique generally used to identify objects, events or observations, which differ significantly from most of the data. There may be several reasons that may cause the presence of an anomaly within a dataset, such as possible errors in measuring instruments or human errors. These can be produced for many reasons, which include possible errors like mechanical faults, measurement instrument error, human error and so on. Depending on the context in which this technique is used, it must present a certain degree of accuracy and rigidity to avoid possible catastrophic scenarios.

This solution can be applied in different contexts. With this work thesis we want to verify the applicability of this technique in a very variable and constantly evolving context such as telecommunications. In particular, we want to try to provide an additional tool to the analysts in the operation to analyze the bankruptcy events that occurred during a mobile call in a specific geographical area. Therefore, this technique will be used to locate areas of the Italian territory in which there has been a significant variation in the number of failures of the voice service, so that analysts can carry out more in-depth analysis in case a situation of this kind.

1.4 THE STRUCTURE

Therefore, this thesis is based on a twofold motivation: the first is to implement some *ad hoc* algorithms that allow the automation of the cataloging operation of the bankruptcy events revealed during the benchmarking missions, while the second motivation foresees the application of the machine learning technique known as anomaly detection, through which it will be possible to verify variations in the number of failed mobile calls in a specific geographic region.

More in detail, thesis work is organized as follows:

- Chapter 2 reports an analysis of the issues related to the CSFB standard. Specifically, after a brief description of the standard, the advantages of its use, the state of the art and the future perspectives are reported, also listing the related problems.

- In chapter 3 the algorithm and the logics at its base that allow the automation of the cataloging of the failed events are presented. In addition, some trends of the output obtained by the algorithm are shown graphically through Tableau, a software used for interactive visualization of labelled data.
- In chapter 4 the technique of anomaly detection is presented, which are the contexts in which it is applied, its characteristics and the working methodology.
- Finally, in chapter 5 the anomaly detection algorithms used, the relative measurements and the graphing of the obtained results are presented.

2 CSFB OVERVIEW

2.1 INTRODUCTION

As first step of this thesis work, was necessary to study the context in which it is focused, in order to better understand the big quantity of data provided by the company.

The context is mainly focused on the telecommunications world, in particular on the mobile voice service provided by different operators working on Italian territory. Operators, not only in Italy but all over the world, look for the latest technologies and facilities to attract more customers. The LTE technology, as already introduced, is the standard currently available on the most advanced and powerful market for wireless communication of mobile devices. Thanks to a full IP infrastructure, it can guarantee very high data transfer peaks and low latency, but also the possibility to use voice and message services directly on the 4G network. Regarding this last point, given the impossibility of having in the short term an infrastructure of this type available anywhere, it was thought of a temporary solution that could guarantee users with LTE devices the ability to make voice calls and send messages without lose all the advantages offered by this new network. This solution is better known as *Circuit Switched Fallback* (CSFB).

The rest of the chapter gives a technical overview of the LTE architecture and, in particular, of the CSFB solution.

2.2 LTE GROWTH AND CHALLENGES

The Long-Term Evolution, also known as **LTE**, is a data standard for mobile devices with an increased capacity and speed with respect the previous mobile standards. It has been developed by **3rd Generation Partnership Project** (3GPP), a telecommunication organization that also created standards for GSM and UMTS.

It has been launched on the market since 2009, but starting from 2014 it has recorded an ever-increasing distribution, as evidenced by Figures 1 and 2, even exceeding, according to the 5G Americas website [2], the three and a half billion connections in 2018. Figure 1 is kindly taken from 5G Americas web site [2], while Figure 2 from LTE Wikipedia page [3].

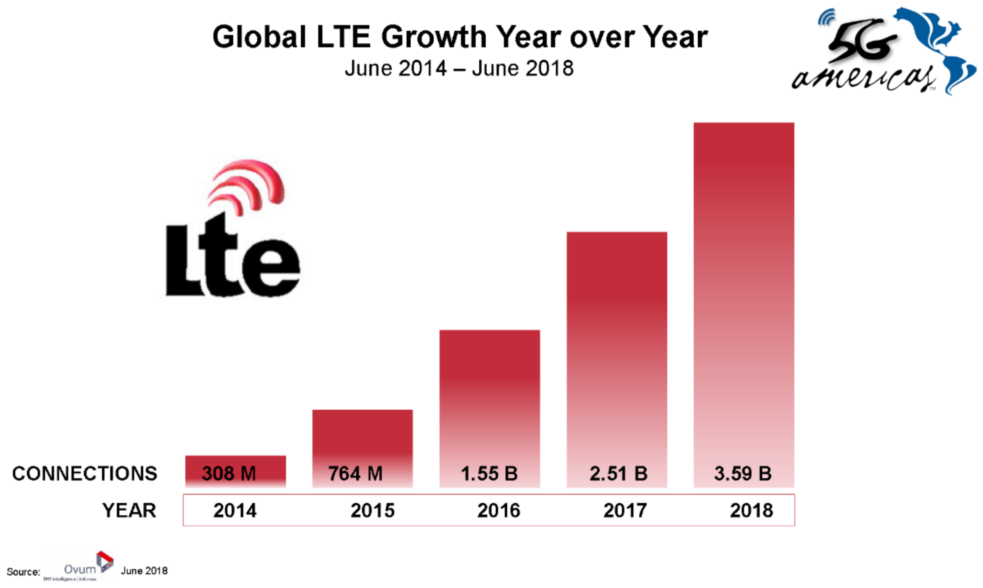


Figure 1. Absolute growth values of LTE connections.

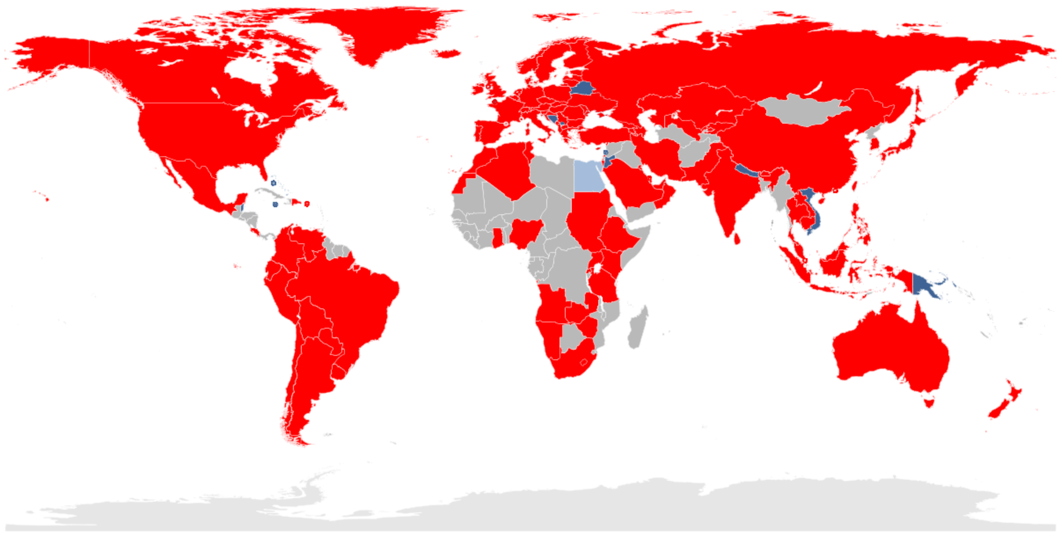


Figure 2. Map of LTE deployment in 2018.

In a full IP network, data is exchanged using a method known as *packet switching* (PS). This method encapsulates the data that must be sent during transmission in small blocks known as packets. Each packet has a header containing different information within it, including the destination to be reached. The packet, therefore, will be able to reach its destination independently of the other ones, thanks to the routing strategies adopted in the network. In this way there is no a unique and unbalanced allocation of the resources and different communications can take place at the same time. Obviously, the channel will have a maximum load capacity, beyond which there will be a problem of congestion and therefore lower performance in terms of transfer speed and latency. Once all the packets arrived at destination, these will be reunited together, thanks to the sorting dictated by a specific field within each one's header.

This methodology differs completely from the other one used in 2G / 3G legacy networks. It is known as *circuit switching* (CS) and aims to allocate all available resources for communication between two nodes, thus forming a circuit. The problem of congestion in this case is less crucial, but the poor sharing of resources lowers significantly its performance. In the 2G legacy networks, voice calls are made practically only on circuits, while in 3G legacy networks these can be made using the CS domain, but also through *OTT (Over The Top)*, a solution quite similar to that designed for VoLTE technology. However, this last solution doesn’t guarantee the QoS requirements to ensure a good communication.

In Figure 3, taken from telecomhall web site [4], is shown a generic topology for 2G, 3G and 4G technologies with their main components and connection among them. Each of these technologies have a different architecture.

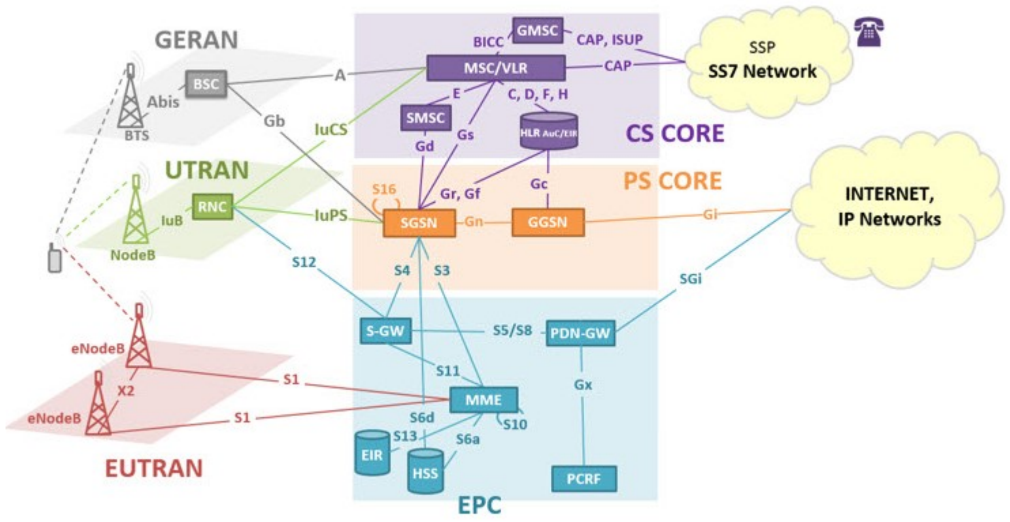


Figure 3. Generic 2G – 3G – 4G network Topology

GSM EDGE Radio Access Network (GERAN) is a radio access network architecture that, through the base station controllers (BSC) allows the connection to the network and therefore to make or receive voice calls directly through the PSTN or Public Switched Telephone Network. The network is connected directly to the UTRAN (UMTS Terrestrial Radio Access Network) core network, which is able to manage both communications (CS and PS). These two architectures represent the 2G (GERAN) and 3G (UTRAN) legacy networks.

Then we have the EUTRAN (Evolved Terrestrial Radio Access Network) architecture, which is standardized for LTE connection. As we can see, there is not direct connection between EUTRAN and GERAN network cores. This means that is impossible to establish directly a voice call being in the LTE network.

The solution adopted to overcome this problem is called **Circuit-Switched Fallback** (CSFB). With CSFB, when a device operating in LTE mode receives an incoming call, the LTE network pages the device. Once device receives this request, it responds with a special service message to the network, starting so the move down operation (fall back) from LTE to 2G/3G legacy networks. The same principle is valid also for outgoing calls, as we will see later.

2.3 VOICE COMMUNICATION SERVICES EVOLUTION

Expectation of voice traffic over LTE handset is growing even more as infrastructures provide a higher LTE availability. Users require even more advanced mobile experience and higher quality of services, characterized by fast operations and high availability. To deal with this continuous increase of quality demand, voice service has evolved over the years. Voice evolution can be characterized into three major phases, as summarized in Figure 4 [5].

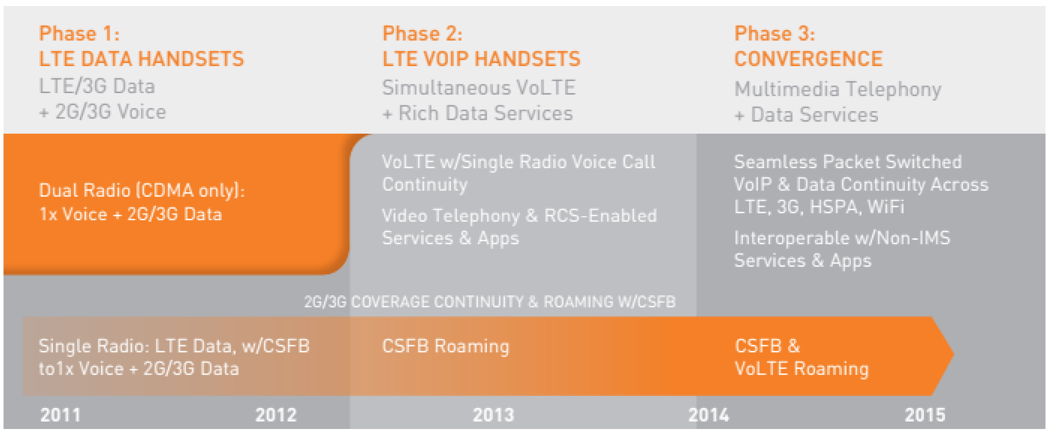

Phase 1:
LTE DATA HANDSETS
LTE/3G Data
+ 2G/3G Voice
Phase 2:
LTE VOIP HANDSETS
Simultaneous VoLTE
+ Rich Data Services
Phase 3:
CONVERGENCE
Multimedia Telephony
+ Data Services
Dual Radio (CDMA only):
1x Voice + 2G/3G Data
VoLTE w/Single Radio Voice Call Continuity
Video Telephony & RCS-Enabled Services & Apps
Seamless Packet Switched VoIP & Data Continuity Across LTE, 3G, HSPA, WiFi
Interoperable w/Non-IMS Services & Apps
2G/3G COVERAGE CONTINUITY & ROAMING W/CSFB
Single Radio: LTE Data, w/CSFB to1x Voice + 2G/3G Data
CSFB Roaming
CSFB &
VoLTE Roaming
2011
2012
2013
2014
2015

Figure 1. Phase of voice service evolution.

In the first phase of 2011, the CSFB technology is introduced into the global market, where the voice and message services are completely managed by the CS networks, while the data transfer takes place through LTE PS network. In 2013 the VoLTE technology was introduced, with the aim of including those services typically performed by CS networks in PS one, thanks to the construction of a full IP ifrastructure. Other IP multimedia services were introduced, like video telephony, HD Voice etc.

With the third phase, a convergence operation is implemented to ensure greater coverage of the services provided.

2.4 How does the CSFB work?

Figure 4 reports phase evolution until 2015. Nowadays, VoLTE development has been completed, but topologies completely based on LTE infrastructures with full IMS are not already available everywhere. So, until this reality doesn't come, to use voice services and SMS delivery is necessary to pass from the LTE network to the legacy one.

These two networks, 2G/3G legacy and LTE, are able to co-exist in mixed networks, staying specifically between the mobile customer's User Equipment (UE) and the common core network [5]. In Figure 5 [5] is illustrated a simplified topology of mixed networks reporting the main actors that serve the UE, depending on the network in which it is located.

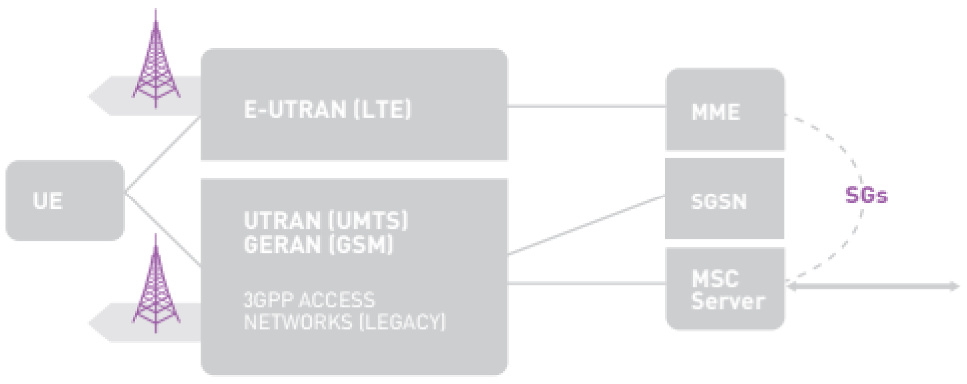


Figure 5. EPS and legacy 3GPP packet core network.

From the image above, we can see the presence of four main actors. The first is the UE (or User Equipment) that represents the user's device that intends to connect to a network in order to use its services. At this point different actors come into play depending on the network in which the EU is operating. The MME (Mobility Management Entity) is the one that serves the devices on the LTE network as regards data transfer (as we have already seen, it is not currently possible to make voice calls on the LTE network with the CSFB), while the SGSN (Serving GPRS Support Node) serves the EU on a legacy network (2G/3G) regarding the data connection, while the MSC (Mobile Switching Center) Server provides voice and messaging services. MME and MSC are connected to each other through a dedicated interface (SGs), through which the switching operation takes place. It allows also the delivery of CS pages through LTE access without leaving the LTE network. In this way, when a device connected on LTE network receives an incoming call, a mobile terminating (incoming) CS voice call triggers a page via LTE to the user’s device, as shown in Figure 6 [5].

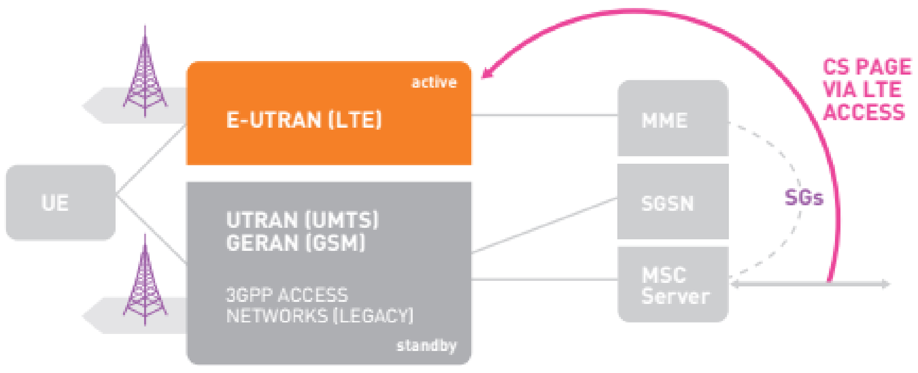


Figure 6. Incoming voice call: CS page via LTE.

At this point, once the page is received, switching from LTE to legacy network can occur as shown in Figure 7 [5]. The connection with the LTE network is interrupted, or better it is put on standby, and therefore the mobile device it is connected to a cell for the 2G / 3G connection.

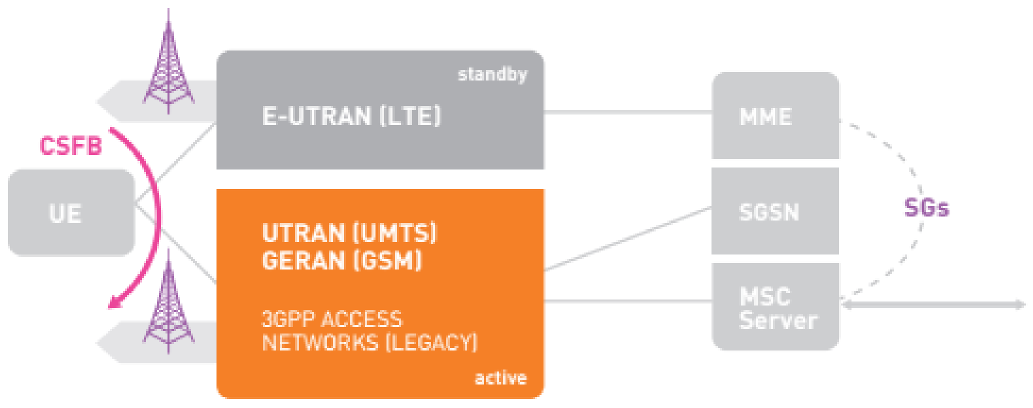


Figure 7. Circuit-Switched FallBack

Once the use of the service has been completed, then the call is over or the message has been sent, the reverse operation is carried out, which then allows returning to the starting LTE network, as shown in Figure 8 [5]. It follows the same principle of the previous one, except for the paging step, which in this case is not needed.

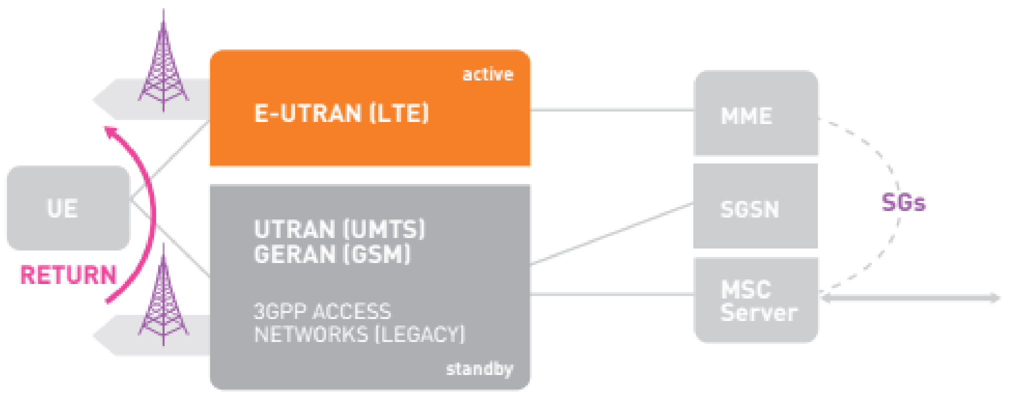


Figure 8. Return to LTE after voice call.

2.5 CSFB CHALLENGES

Switching from one network to another poses many challenges in terms of efficiency and reliability. Generally, the acquisition of a cell for 2G / 3G legacy networks is done through two different operations: *handover* or *redirection.* The first operation, *handover*, refers to a process in which a phone call or data session is transferred from one channel to another one. This usually happens when the user is moving and when the coverage of the cell to which it is hooked becomes increasingly scarce, then he tries to link himself to the nearest one or, in any case, to that one that allows him to have a greater power of the signal. During the handover procedure, the selected channel, or cell, is prepared in advance and the device can enter directly in connected mode. Before it, is necessary some IRAT (Inter-Radio Access Technology) measurements to check the power signal strength of the selected cell.

The redirection procedure, on the other hand, involves hooking up to a cell based on a list of frequencies indicated by the device that wants to switch. Once the hooking procedure has been completed, the device can complete the voice or messaging service. For this kind of procedure, usually IRAT measurements are not required. Generally, a solution of this kind is preferred because it has operative times with respect to the handover procedure, even if the latter is more reliable.

CSFB based on redirection has variations with differing call setup speeds [5]:

- ***Release 8 Release with Redirection*** - This is the first version based on the redirection procedure, where the device that wants to access the selected cell must first read the System Information Block (SIB) messages, labelled as "UMTS SIB Read and Camp time". SIBs are the dynamic part of the System information (SI) and are mapped on RRC SI messages (SI-1, 2, 3, 4, 5, 6, 7, 8, 9, 10 and 11) over DL-SCH (Downlink Shared Channel), a transport channel used for downlink system information, and transmitted using PDSCH (Physical Downlink Shared Channel), the physical channel that carries the DL-SCH coded data, at periodic intervals [6].
- ***Release 8 Release with Redirection - SIB Skipping (3G)*** - same procedure of the basic version, but here not all the SIBs are read, only the mandatory ones like 1, 3, 5 and 7, skipping so all the other SIBs prior access. The remaining blocks are acquired through the DMCR (Deferred Measurement Control Reading).
- ***Release 9 Enhanced Release with Redirection - SI Tunnelling*** - with this version SIBs messages are included in the redirection message sent to the UE during the hook operation to the cell.

Nowadays, *Release 8 Release with Redirection-SIB Skipping* is the preferred solution thanks to the reduced call setup time, reliability a simplicity, although there are often delays that can significantly damage the quality of the user experience.

Different studies and measurements [5] have been conducted both for mobile originated (outgoing) and mobile terminated (incoming) voice calls. For outgoing calls, 3G has good performance using both handover and redirection procedures, while 2G works well only with redirection-based Release 9 SI Tunnelling. Incoming calls have obtained similar results, where the best performance has been reached with handover-based call setup, just slightly faster than redirection-based Release 9 SI Tunnelling.

Another fundamental aspect is the time of data interruption, that is the time that the data service is interrupted during the switching from the LTE network to the legacy one.

The interruption time strongly depends on the kind of mechanism adopted, as reported in Table 1. Values reported in this table are taken from Qualcomm white paper [5].

Table 1. Data interruption time measurements.

	Handover	Redirection		
	Release 8 / Release9	Release 9	Release 8	
		SI Tunnel	Skip SIBS	Basic
Handover	0,3			
RRC Release		0,2	0,2	0,2
Acquisition on Utran		0,2	0,2	0,2
Read MIB & SIBs			0,4	2
Camp on Cell		0,1	0,1	0,1
Connection Setup		0,3	0,3	0,3
Optional RAU Procedure		4	4	4
Total	0,3	4,8	5,1	6,8

As can easily be seen from the table, the handover procedure allows to have break times much lower than those recorded with the redirection procedure.

3 Data Preparation - Correlator & Comparator

3.1 Introduction

Once that the study phase about the main technologies and protocols that characterize the mobile voice service has been completed, then was possible to start with the data analysis. Data provided by Reply is the result of a complex operation of filtering, cleaning and adjustments of logs obtained from "benchmarking missions". Specifically, these missions consist of drive car fitted out with a specific benchmarking equipment that collect logs about performance, reliability and many other parameters useful for the analysists. Thanks to these logs, the expert staff of analysts in Reply can determine coverage's quality of a specific zone, providing feedback to the telecommunication operators. What the analysts study is how a mobile call evolved and the causes of a fault, where for fault we mean a mobile phone call failed, and their objective is to discover the main cause that produced that specific fault.

A proprietary software helps this analysis operation, but sometimes it is a very long and tough procedure and so it requires a huge quantity of resources in order to be carried out on time with the deadlines fixed by the clients, in this case the telecommunications operators. In this sense, was necessary to create an automated algorithm such that was possible to speed up and, in some way, optimize this analysis operation.

In this chapter will be analysed, first of all, the main procedures that allow the logs to be carried out, describing then the algorithm implemented to obtain this automation and different comparisons with old mission already analysed by the staff, in order to verify the output fairness. And the end, we will have also some trend consideration about the analysed missions.

3.2 BENCHMARKING TESTS

Subscriber perceived quality, known also as Quality of Experience or QoE, is one of the aspects most taken into consideration by mobile operators, and for this reason they are constantly looking for possible optimization of the services provided.

Benchmarking tests are the optimal solution to quantify possible improvements needed to optimize these services and to pinpoint specific areas for service and network related KPIs where investments, i.e. further optimization activities or network improvements, should be targeted.

This operation is usually delegated by the telecommunication operators to consulting companies like Reply. In order to do that, Reply use *Rohde & Schwarz Mobile Network Testing solutions.* These solutions provide all the hardware and software to be able to conduct as efficiently as possible and with the best quality all the analyzes necessary to customers in order to optimize their network infrastructure. In Figure 9, kindly taken from company web site [7], are reported some of the products developed by *Rohde & Schwarz*.

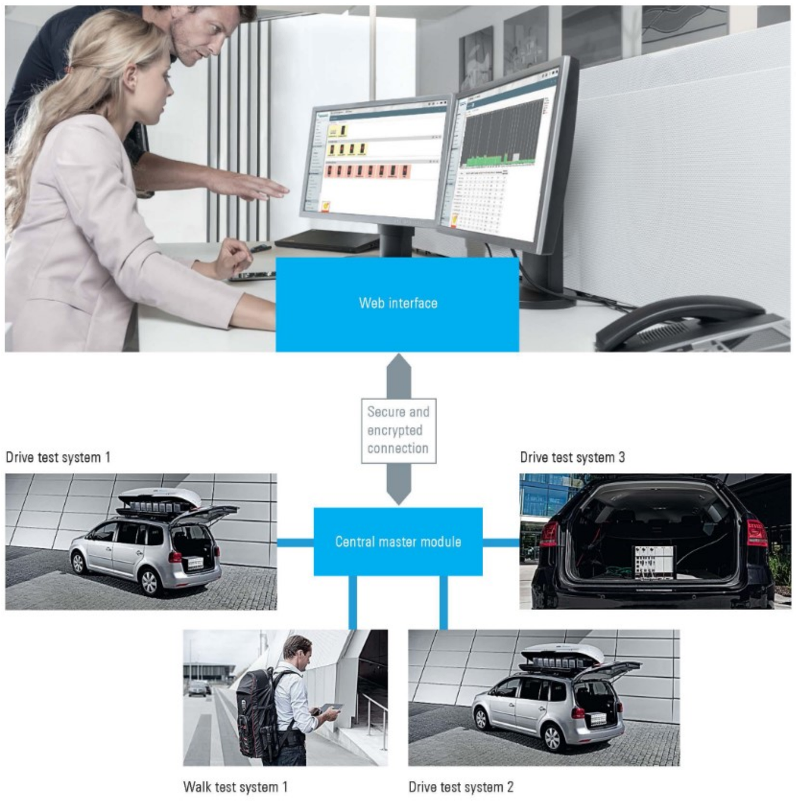


Figure 9. Smart architecture example

Logs are collected through specific equipment that, thanks to dedicated software, a wide range of latest generation smartphones and continuous remote controls, makes them reliable and easily reproducible, then easy to interpret for customers who want to optimize services to be supplied to its customers.

The specific product used for this purpose is **SmartBenchmarker**, a new software platform from Rohde & Schwarz mobile network testing for benchmarking tests. It is the technological evolution of the *SwissQual Diversity* and *NQView platforms* and allows the seamless configuration. It offers complete fleet management, advanced alarming and monitoring functions, providing also more flexibility when conducting drive tests and performing accurate and advanced benchmarking in line with the latest standards and methodologies.

SmartBenchmarker allows users to remotely control one or more benchmarking systems. Via a secure and encrypted connection, a test supervisor can log in to the system and simultaneously manage the vehicle and monitor multiple devices (smartphones and scanners). In Figure 10 [7]is shown a general SmartBenchmarker setup.

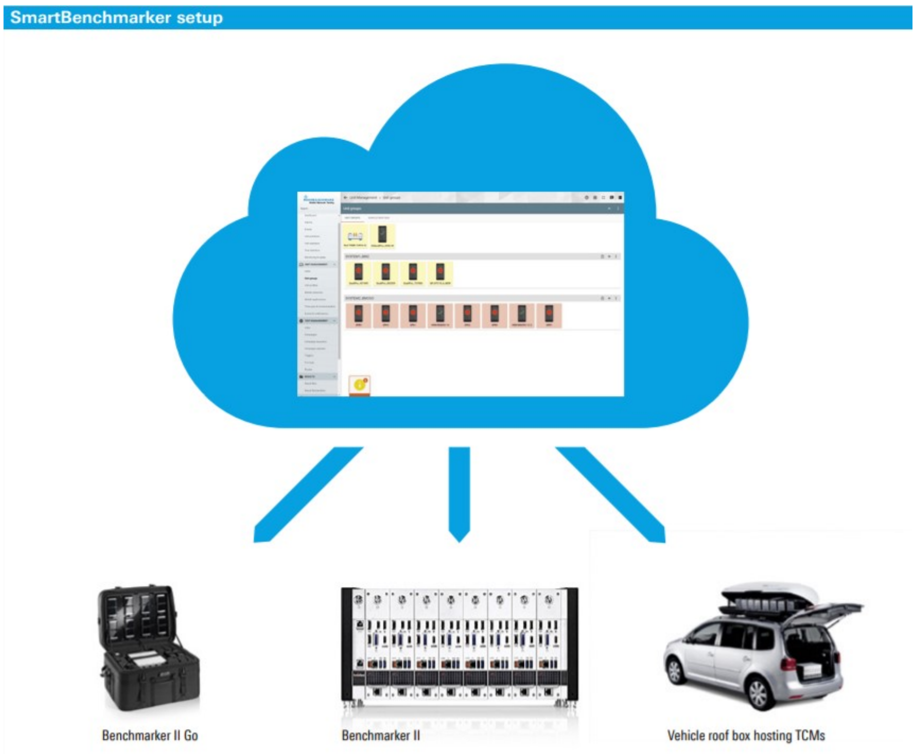


Figure 10. SmartBenchmarker setup

3.2 DATA EXTRACTION

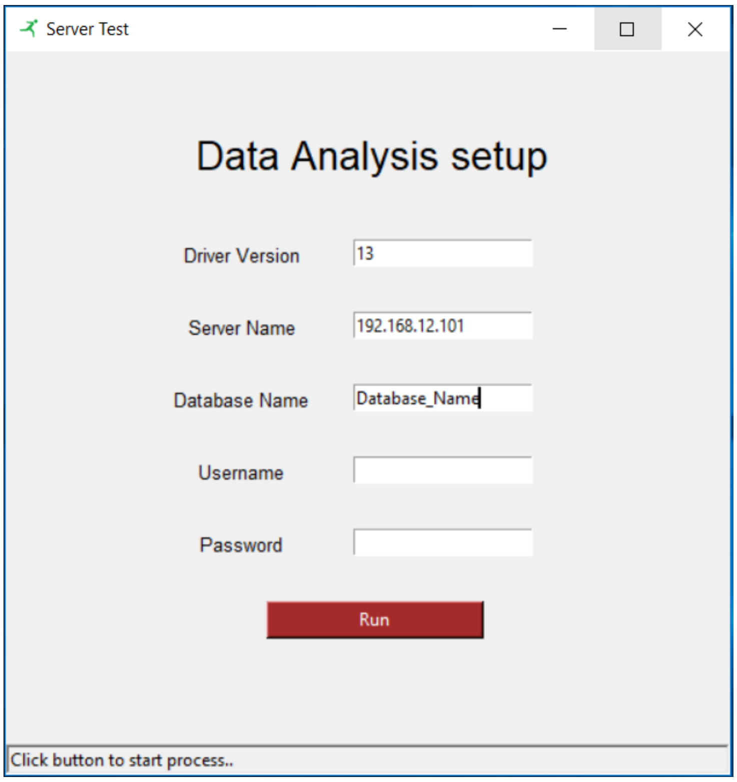
All the logs collected by this advanced benchmarking equipment are also stored in a SQL Server owned by Reply. A **Microsoft SQL Server** is a relational database management system developed by Microsoft. The reasons that led the engineers to make this choice of storing these logs on a SQL Server are first of all availability, but also scalability, high-performances, security and lower cost of ownership.

In order to access to the Database is necessary to perform a remote connection to the internal server and use specific tool, in our case Microsoft SQL Server Management Studio, to obtain the desired table containing all the information. As said in the introduction of this chapter, these logs are 'modelled', so what we obtain performing a simple 'SELECT' SQL query is the result of these information passing through thousand and thousand lines of SQL code, where they are filtered, collected, joined with other customized tables and so on. At the end of this long operation (sometimes it can require different hours to be correctly performed), what we obtain is a table containing the main causes of the faults that characterized the missions.

This is a very crucial point, also for the following steps, because all the analysis are done only on faults and on their causes, we don't have data with mobile calls that went well or failed, but only failed ones. Faults are mobile phone call that failed, so that was not possible to establish a connection between two devices, identified with a unique identification. We only know that these happened at a certain time and for a single fault there can be many possible causes. Our objective is to find out the main cause of the single fault, as the analysts do, but in an automatic way.

The procedure necessary to obtain these tables, as described before, is to perform these complex SQL queries and at the end of those execute a 'SELECT' on the final table, called *Causes_EXT*. Due to the fact that sometimes the client, in our case mobile operators, want to analyze internally these tables, a simple software was developed in such a way to hide all these proprietary lines of SQL code. This software has been developed in Python, an interpreted, high-level, general-purpose programming language. For this task different libraries have been used, like *tkinter* for the GUI (Graphical User Interface) and *pyodbc*, a python library used for Database connection.

In Figure 11 is reported a screenshot of the software's interface.


Server Test

Data Analysis setup

Driver Version: 13

Server Name: 192.168.12.101

Database Name: Database_Name

Username:

Password:

Run

Click button to start process..

Figure 11. Software interface for log selection

The interface presents different fields that must be filled with the correct server's proprieties, like the Driver Version, Server Name and Database Name. There are also Username and Password, but these last two are not mandatory. If these are not filled, connection is automatically performed through *Windows authentication*. Once that this tool is executed properly (otherwise a message error will appear), the output is an excel file with the following columns:

- **Provider** or **HomeOperator**: mobile operator holding the sim card present inside the smartphone during call failure.
- **Mission**: identification of the mission. It indicates the region where the benchmark campaign has been done.
- **SessionID_Number**: unique identifier of the call. Is possible to see this identifier as the failure identifier because, as said before, we deal only with failures, so if that identifier is present in the table, this means that the call failed.

- **SessionID_MO**: unique identifier of the call originator.
- **SessionID_MT**: unique identifier of the call receiver.
- **ASideDevice**: it indicates the device (smartphone model and firmware) used by the call originator.
- **BSideDevice**: it indicates the device (smartphone model and firmware) used by the call receiver.
- **Failed_side**: it indicates which is the side that causes the fault, originator or receiver.
- **Phase**: phase where the fault is happened.
- **Category**: category where the fault is happened.
- **RootCause**: root cause where the fault is happened.
- **RootSubCase**: It's an additional parameter where analysts can put some measurement information, like the interference, RSCP, signal-noise ratio and so on.
- **AdditionalInformation**: This is an optional field where analysts usually put some comment about the fault.
- **CauseID**: identification of the cause.
- **_TimeStampEvent**: date and time when the fault has been registered.
- **_Weight**: is the weight that we give to the fault event. It's correlated with CauseID.

2.4 DATA MINING – CORRELATOR

Final output, as said before, is composed of a huge number of rows as much as the number of faults registered during the single mission and the main causes recognized for a specific fault by the SQL algorithm. The main motivation because a single fault usually present more than one possible cause is due to CSFB's protocol stack that has been developed and updated during all these last years. Figure 12.a and 12.b show how works the overall procedure of connection through the CSFB.

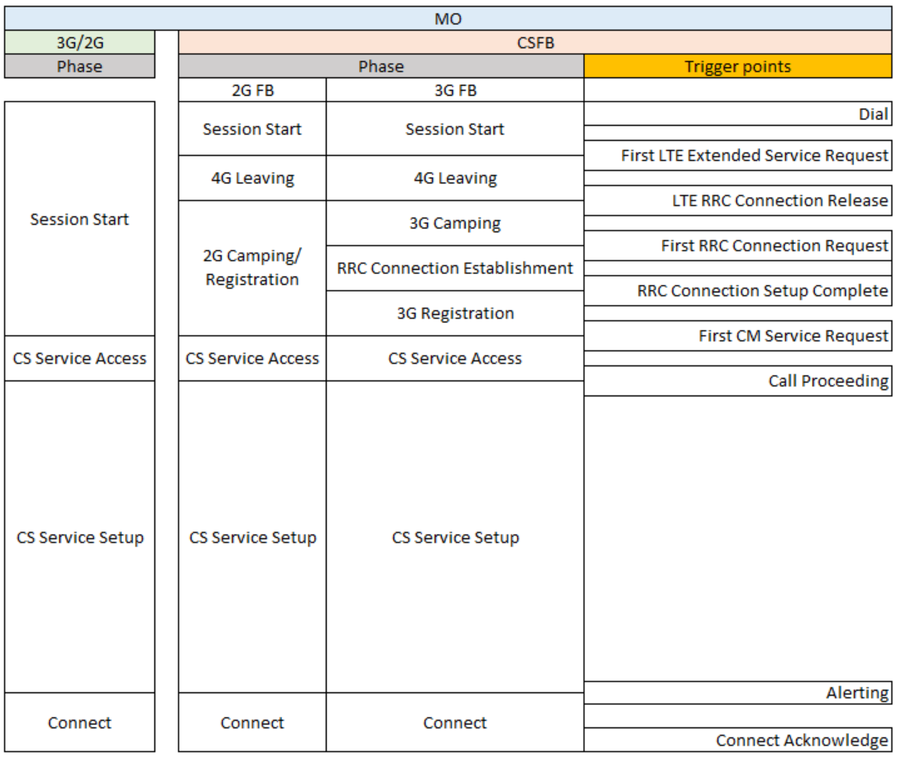


Figure 12.a CSFB procedure of call's originator (MO).

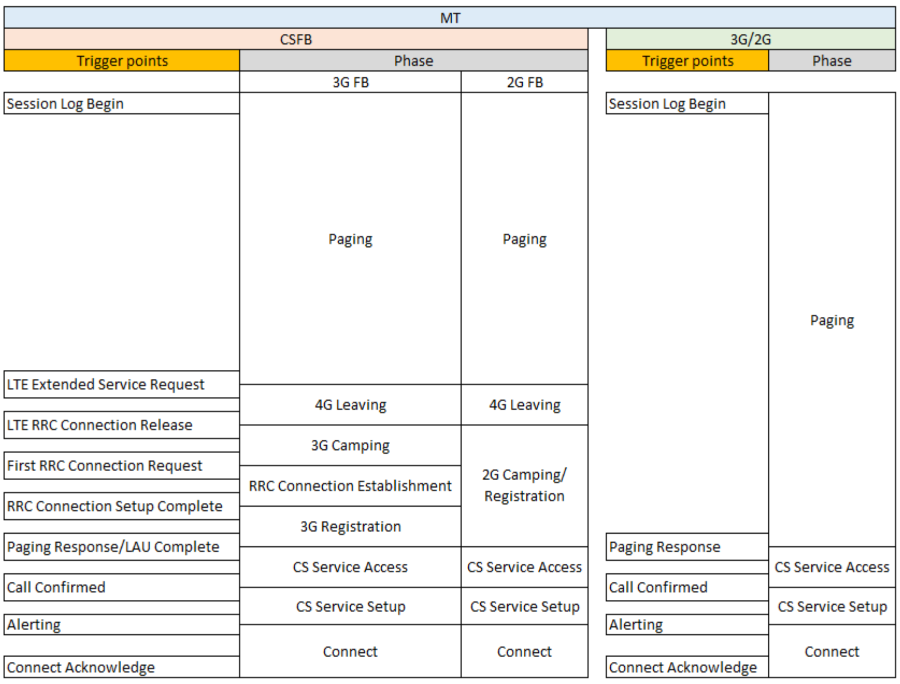


Figure 12.b CSFB procedure of call's receiver (MT).

The overall procedure is separated in two distinct steps, the first one that concerns the MO side (call originator), Figure 12.a, so the smartphone that is starting the mobile call, while the second one is related to the MT side (call receiver), Figure

12.b, who is receiving the call. How is possible to notice, the whole procedure is divided in different phases that allow the ‘downgrade’ (*fall-back*) from LTE to 3G/2G legacy domain. Obviously, we have different phases if the conversion is done toward 3G or 2G.

When a fault occurs, its main cause can be localized in one of these phases above-reported, but sometime is not the first one registered in terms of time. This because the protocol has been optimized during all these years and so it presents different recovery procedures that are able to restart procedure from the last completed phase before that, for example, some interference disrupted the signal or maybe for a poor coverage problem. This means that a failed call could have been recovered one or two times, and so the errors are recorded, before being classified as failed, but the main should not be found in those errors. Once that a call is classified as ‘failed’, that means that was no even more possible recover that, so a series of phases are registered in which an error occurs, starting from the last phase that did not complete its own procedure. A cause, so, is a phase that has not been completed.

This is the main reason that explain the presence of so many causes registered for one single fault. Obviously, not all the causes registered must be considered, this because some of those are just a result of the state-of-art of the build-up network (the previous phase was not correctly completed, so this produce an error). So, they are recognized as errors, but they are not candidate to be recognized as the failure’s main cause, and for this reason, generally, these are filtered out by the algorithm.

Now is necessary to find the main cause of the fault. To do this, the approach adopted was to use a secondary file where are stored many information about the ***CauseID***, one of features present inside the *Cause_EXT* file. Among the different information present inside this file, called *CauseDescription*, one of the most important one is the **Weight**. What we want to do, so, is to give to each of cause a certain weight such that the cause with the highest weight is recognized as the main cause. If two or more causes have the same weight, the main one is the first that is appeared.

Other two very important features contained inside the *CauseDescription* file are **TDR** and **TDL**. These two parameters give information about the time interval of observation that must be considered for each **CauseID**. So, for each cause we define a temporal interval in which we see what happened before, TDL, and after it, TDR, in order to see how causes influence each other. In Figure 13 is shown an example of this observation interval, where the origin is the cause's timestamp.

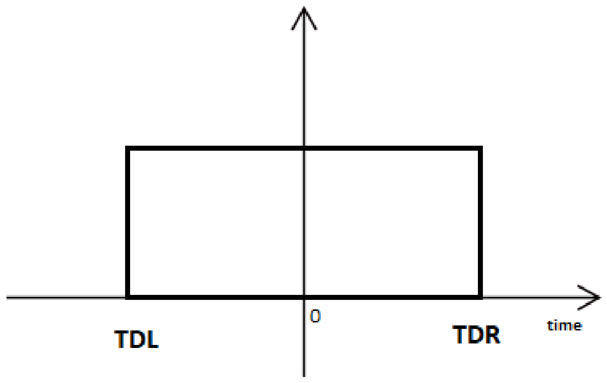


Figure 13. Example of TDR and TDL

Some causes have a wide interval, like a missing paging, and other ones with an interval equal to zero, like system errors (i.e., wrong telephone number, no sync, dial too late etc.), which are the more restrictive, so that we know that if they appear, they are for sure the main cause. Thus, if at a certain point a cause hasn't any influence on the following one, so the next cause happened too late with respect the TDR of the previous one, we only consider those that were candidates to be selected as the main cause, those therefore whose observation interval was part of the others, like shown in Figure 14.

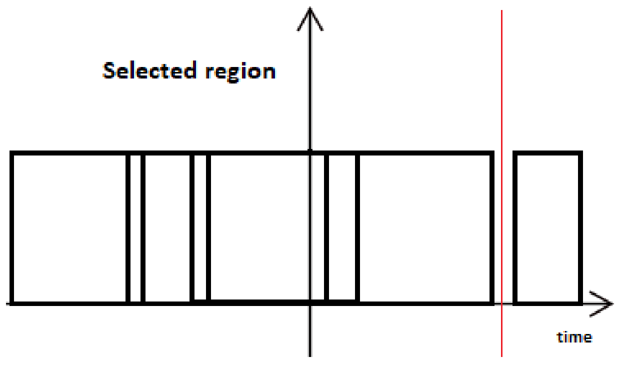


Figure 14. Example of overlapped observation interval. The red line marks the region between possible main causes and causes that must not be considered anymore for that fault.

Among these we select that one with the highest weight, and if two or more have the same weight, we choose the one that comes first in time.

Time, like weight, is set accordingly with the matured experience on this kind of analysis, so it required a lot of weeks to be correctly set, thanks to different interviews done with the analysts. The following lines of code are the implementation of the logic just described. Code is written in Python 3.7.

```
file = pd.read_excel('Causes_EXT.xlsx')
desc = pd.read_excel('CauseDescription.xlsx')
session_id = file['SessionID_Number'].unique()
differences = []
arr = []
arr_fin = []
ind = -1

for id in session_id:
    temporary = file.loc[file['SessionID_Number'] == id]

    if temporary.shape[0] is 1:
        arr_fin.append(temporary.values[0])

    else:
        time_stamp = temporary['_TimeStampEvent'].values
        data = np.array([temporary['_Weight']])

        for index, row in temporary.iterrows():
            ind += 1
            data_copy = data.copy()
            time_stamp_copy = time_stamp.copy()
            temp = row['_TimeStampEvent']

            for x in range(len(time_stamp_copy)):
                diff = pd.Timestamp(time_stamp_copy[x]) - temp
                differences.append(diff.total_seconds() * 1000)

            df = desc.loc[desc['CauseID'].isin([row['CauseID']])]
            TDL = df['TDL'].values[0]
            TDR = df['TDR'].values[0]

            for i in range(len(time_stamp_copy)):
                if differences[i] < -TDL or differences[i] > TDR:
                    data_copy[0][i] = 0

            differences.clear()

            if 0 not in data_copy:
                max = np.argmax(data_copy)
                final = temporary.values[max]
                arr_fin.append(final)
                arr.clear()
                ind = -1
                break

            if ind is not temporary.shape[0] - 1:
                if data_copy[0][ind + 1] == 0:
                    if len(arr) == 0:
                        final = temporary.values[0]
                        arr_fin.append(final)
                        arr.clear()
                        ind = -1
                        break
                    else:
                        arr.append(data_copy)
                        temp_max_arr = []
                        [temp_max_arr.append(np.max(arr[x])) for x in
range(len(arr))]
                        final =
temporary.loc[temporary['_Weight'].isin([np.max(temp_max_arr)])].sort_values(
                            '_TimeStampEvent', ascending=True).values[0]
                        arr_fin.append(final)
                        arr.clear()
                        ind = -1
                        break
                else:
                    arr.append(data_copy)
            else:
                arr.append(data_copy)
                temp_max_arr = []
                [temp_max_arr.append(np.max(arr[x])) for x in range(len(arr))]
                final =
temporary.loc[temporary['_Weight'].isin([np.max(temp_max_arr)])].sort_values(
                    '_TimeStampEvent', ascending=True).values[0]
                arr_fin.append(final)
                arr.clear()
                ind = -1
                break
```

3.5 Data mining – Comparator

Once logic has been implemented properly, fixing bugs and optimizing in terms of computational time, then the comparison step started. In this phase we tried to compare the output of the correlator step with the output of the same mission done by the analysts, considering different missions, in order to evaluate a degree of veracity of the algorithm (this because we want to obtain a result as much similar as possible with that one evaluated by analysts). For degree of veracity we mean the number of matched faults between the two output over the total number of faults.

This confrontation was not so easy as we could expect at the beginning. This because, even if analysts' output is a XLSM file, a macro-enabled spreadsheet created by Microsoft Excel (a macro is a useful feature that functions like a script, which allows users to automate repetitive tasks) where columns are characterized by a drop down menu where they can select only one value among those present in the set, some of the columns' file do not present this function, giving to analysts a higher degree of freedom when they report their final consideration. During years, they tried to find a standard for most of the features that did not present a drop-down menu, like **RootCause**, **RootSubCase** and **AdditionalInformation**, but mobile communication protocols evolve and change continuously, making this standardization operation very tough and complex.

We understood that make a comparison for all the features present in these two files was impossible. Slight string variation, bad-located commas, underscore and so on making the comparison's result insignificant, with a degree of veracity between the 2% and 5%. We so decided to change our approach, where we consider only the most relevant feature. These selected features are: **Phase**, **Side**, **Category** and **RootCause**.

In table 2 are reported the results obtained by the comparison analysis:

Mission	Phase & Side	Category	RootCause	Record analyzed	Matched records	Mis-matched records	Record no present in both	% of matched
mission X (1)	YES	NO	NO	378	300	78	9	79,34%
mission X (2)	YES	YES	NO	378	255	123	9	67,46%
mission X (3)	YES	YES	YES	378	217	161	9	57,41%
mission Y (1)	YES	NO	NO	92	74	18	12	80,43%
mission Y (2)	YES	YES	NO	92	41	51	12	44,57%
mission Y (3)	YES	YES	YES	92	29	63	12	31,52%

Table 2. Results obtained from the comparison operation between the algorithm output and the analysts one.

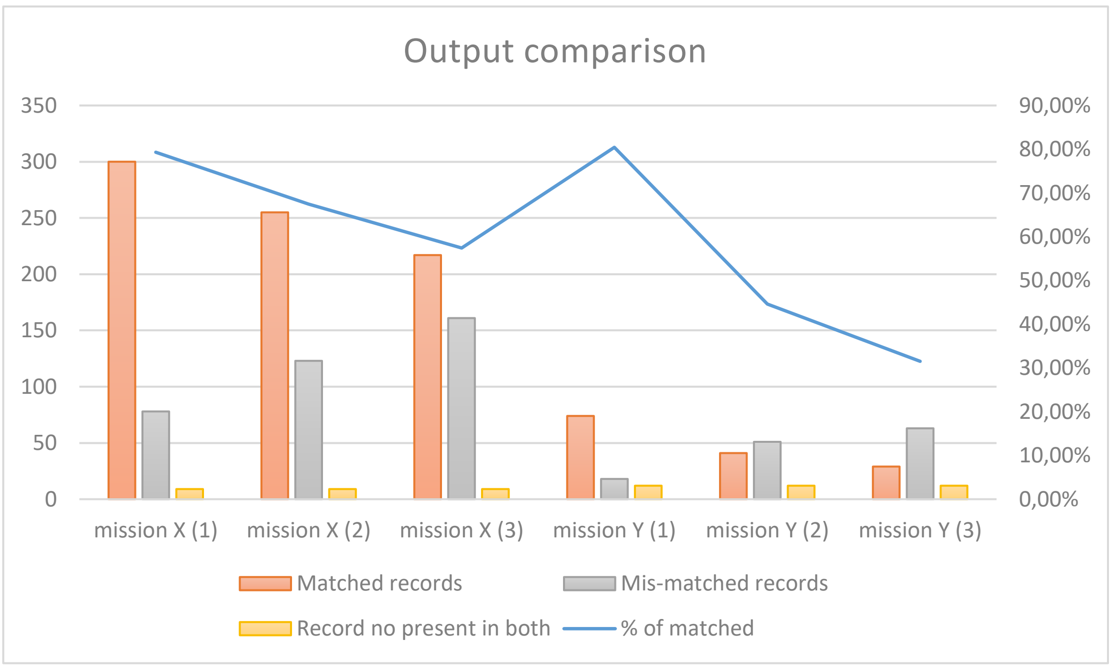


Figure 15. Comparison's result graphication

Comparisons have been done for two different missions: mission X and mission Y (it is not possible to report the correct name of the missions because of the privacy policies inside the company). The first one is mostly located in the south part of Italy, while the second one is related to highway in north and middle Italy. Figure 16 and Figure 17 show the specific location of these two missions.

About the results obtained, two different approaches were used. First, we do a comparison only on **Phase** and **Side**, then we compare also **Category** and **RootCause** in addiction with the first two. It is easy to notice how the percentage of records correctly matched decrease as we consider more parameters instead of the initial two. We can notice a linearity of this decrease as the number of parameters involved becomes larger. Anyway, the algorithm seems working well on a dataset with a considerable number of records, like in the case of mission X, while for smaller missions in terms of size present a higher decrease.

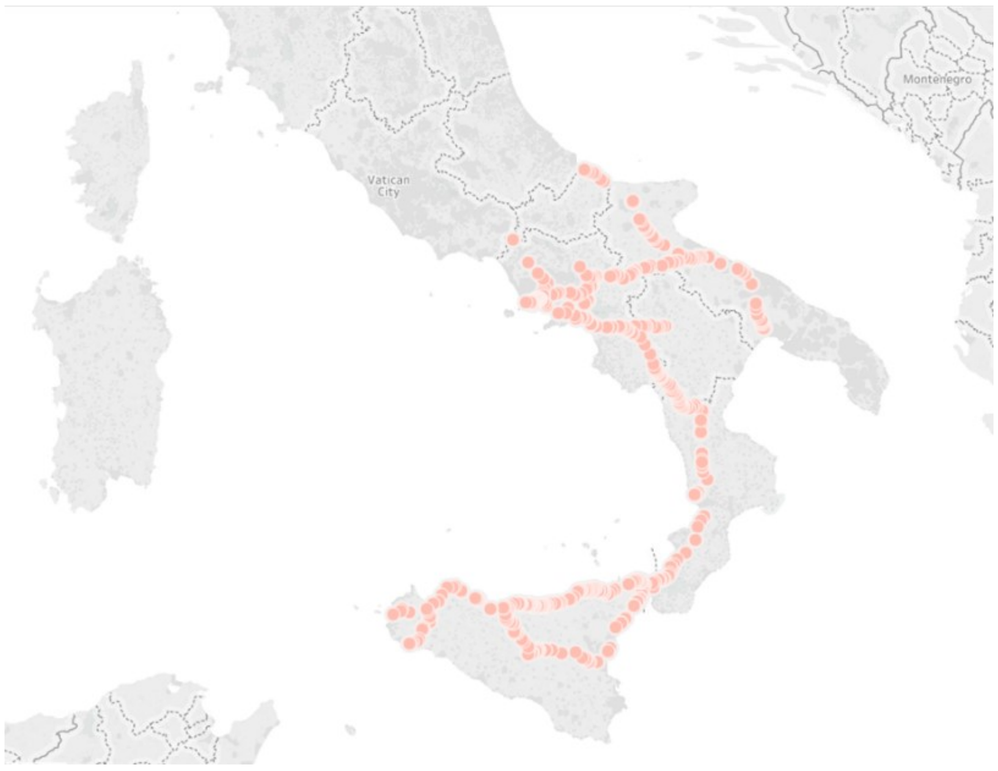


Figure 16. Mission X zone

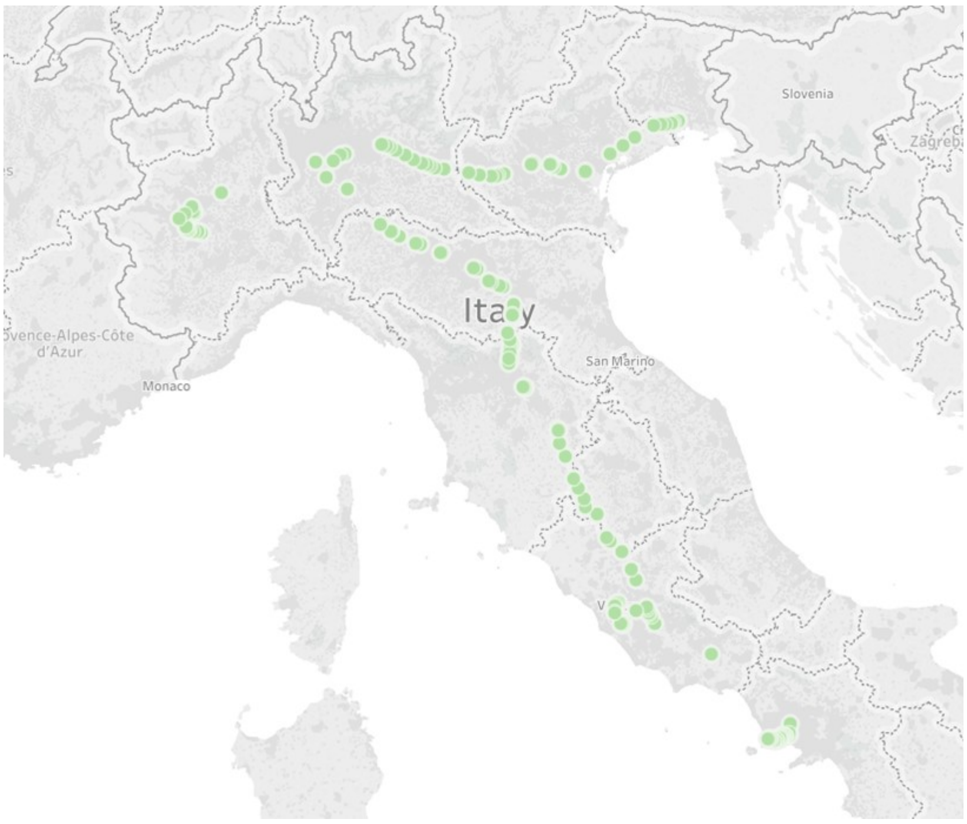


Figure 17. Mission Y zone

However, the percentage obtained considering only **Phase** and **Side** leads to good results in both missions, which are at the end the most interesting parameters requested by the clients, but some optimization and alignment with the outputs evaluated by analysts is necessary in order to obtain a higher level of accuracy from the algorithm.

3.6 TREND RESULTS

Then, once that this degree reached a good value, between 70% and 80% as in the case of Phase and Side, we generated some trend based on these results. Trends were generated considering the most relevant parameters usually requested by the clients. To create these trends, we used **Tableau**.

Tableau is a data visualization software that is used for data science and business intelligence. Tableau can create a wide range of different visualization to interactively present the data and showcase insights, easily understood by anyone.

Tableau also comes with real-time data analytics capabilities and cloud support. For our purpose, we used **Tableau Desktop**.

Tableau Desktop is a business intelligence and data visualization tool. It specializes in transforming usual tabulated data into graphs and representations. With tableau desktop, is possible to deal with real-time data analytics from data warehouses or easily import data multiple sources and integrate them in interactive dashboards.

From Figure 19 to Figure 26, the graphic results of the trend analysis done through Tableau are shown. In particular, the analysis done is a comparison between two missions done in the same zone but in two different period of the year. These missions will be called **mission 1** and **mission 2**, because of the company privacy reasons.

In Figure 18, instead, is shown the zone in which these two missions are evaluated. It includes the main railways of the north-east Italian territory, passing through the regions of Emilia-Romagna, Veneto, Friuli-Venezia Giulia and Trentino Alto Adige.

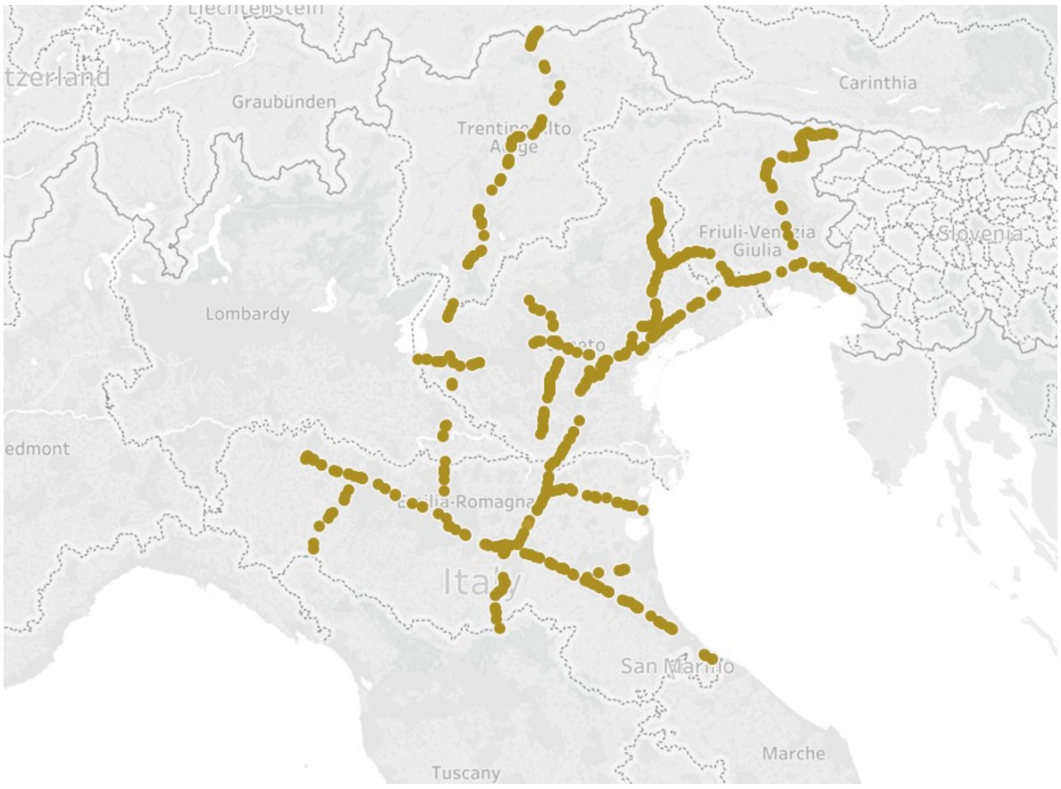


Figure 18. Mission 1's Italy zone.

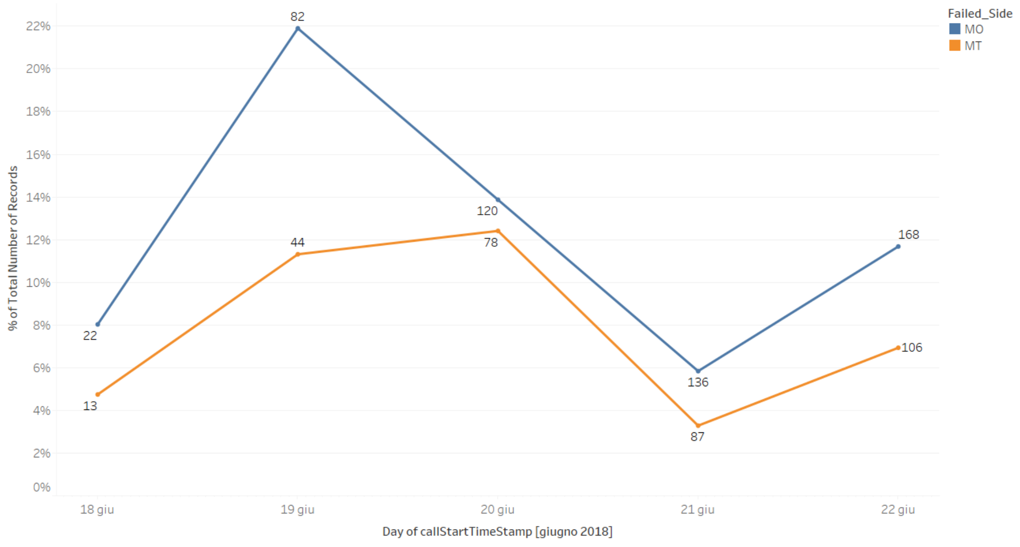


Figure 19. Percentage of Failed_Side over time for mission 1

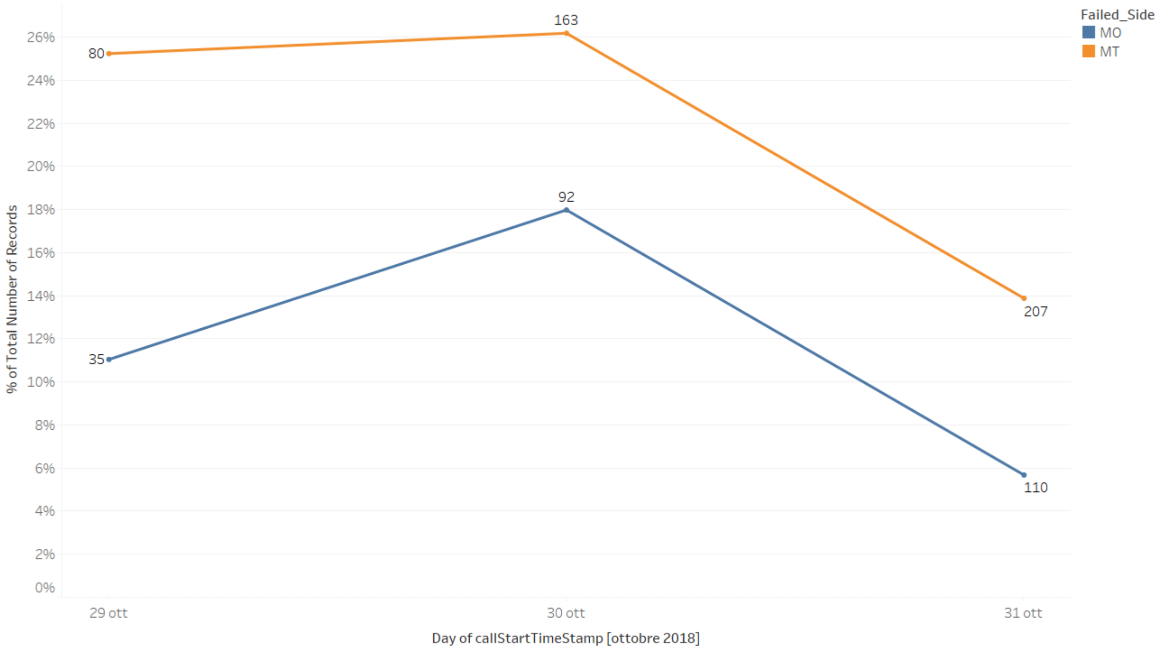


Figure 20. Percentage of Failed_Side over time for mission 2

Figures 19 and 20 show the trend of percentage of total number of records for each day of the mission (we consider callStartTimeStamp feature divided per day). Color shows details about Failed_Side. The marks are labeled by running sum of count of number of records. The data is filtered on callStatus and Mission. The callStatus filter keeps Failed (we are interested only on failed mobile phone calls), while the Mission filter keeps mission 1 for Figure 19, and mission 2 for Figure 20. Views are filtered on Failed_Side, which keeps MO, the originator of the mobile call, and MT, the mobile call receiver.

We can easily notice how these two parameters, MO and MT, are literally inverted in these missions, being MO higher than MT in mission 1 and the opposite in mission 2.

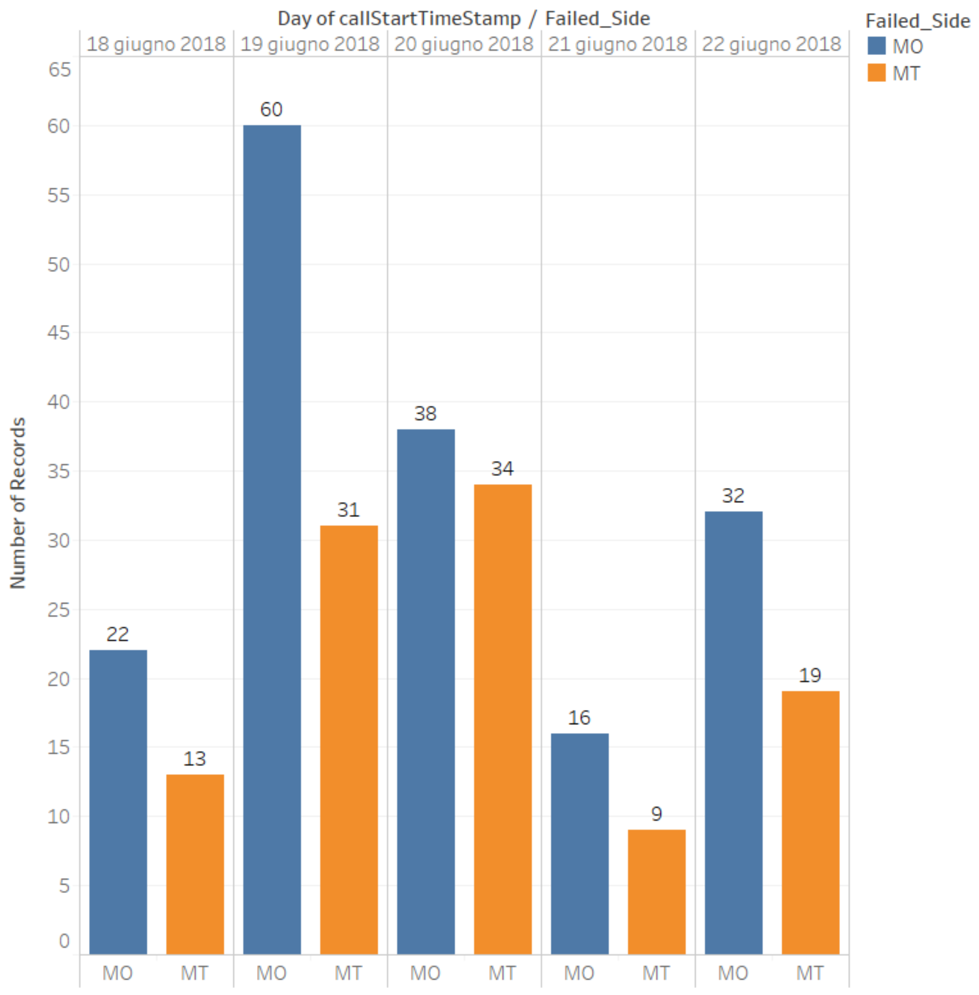


Figure 21. Number of records for each Failed_Side registered during mission's days – mission 1.

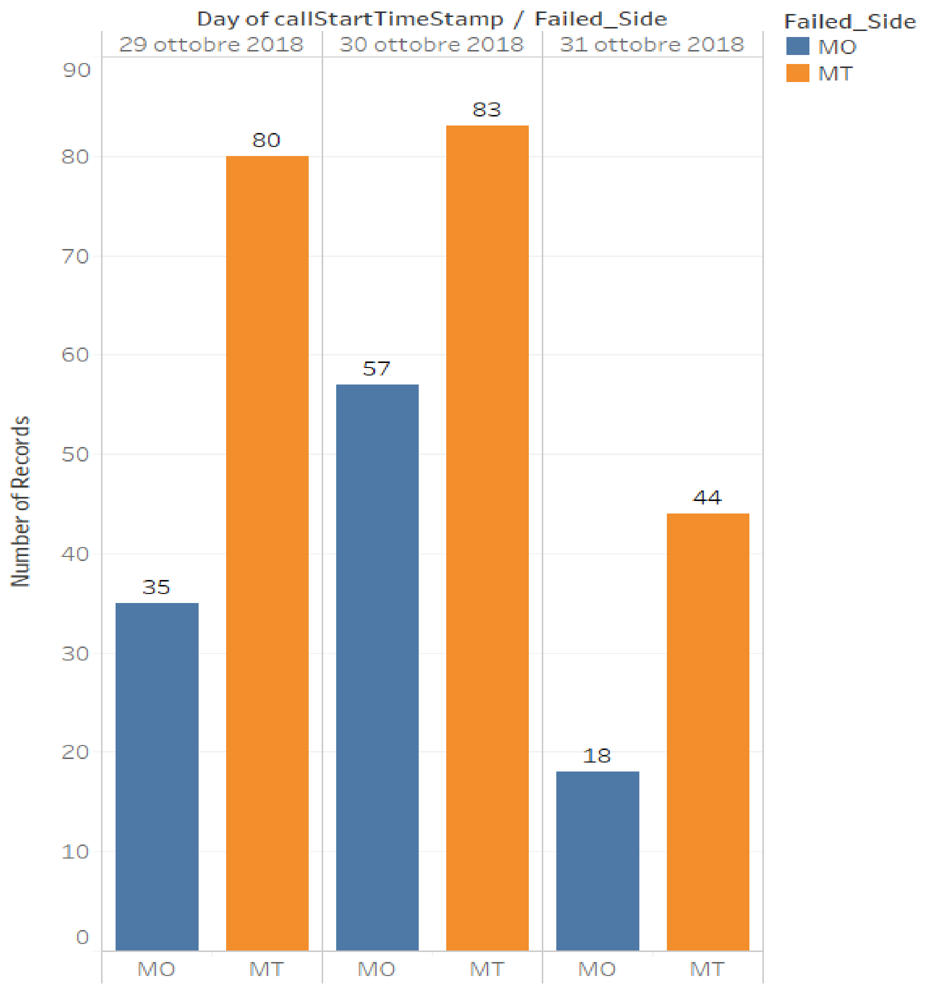


Figure 22. Number of records for each Failed_Side registered during mission's days – mission 2.

Figure 21 and Figure 22, instead, shows the sum of number of records for each Failed_Side broken down by days. Colors, marks and filters are the same seen before for Figure 19 and Figure 20. The main difference of these two last charts, Figure 21 and Figure 22, with respect the first two is that here we look at the number of occurrences of MO and MT divided per days, while in the others is shown the sum of these occurrences along days.

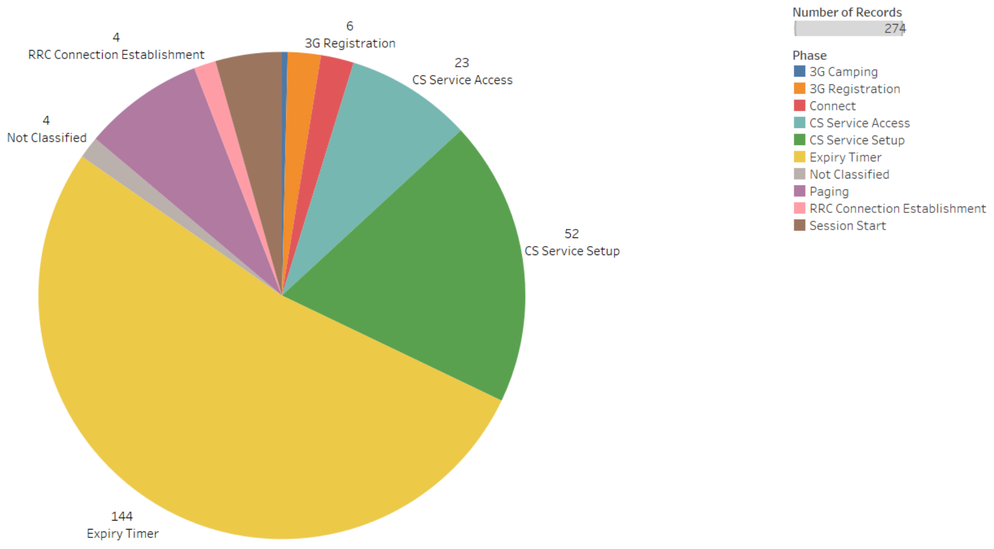


Figure 23. Pie chart of the number of occurrences of each phase registered during the mission – mission 1.

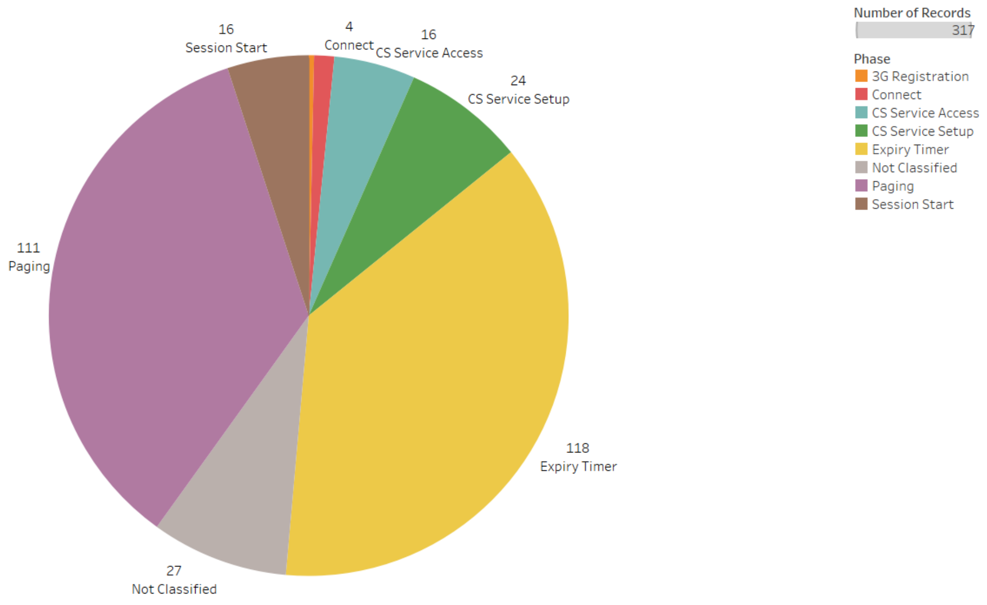


Figure 24. Pie chart of the number of occurrences of each phase registered during the mission – mission 2.

Figure 23 and Figure 24 show the sum of number of records and Phase. Color shows details about Phase, while size shows sum of number of records. The marks are labeled by sum of number of records and Phase. The data is filtered on Mission and callStatus. The mission filter keeps mission 1 in Figure 23 and mission 2 in Figure 24. Also in this case, the callStatus filter keeps Failed records. Views are filtered on phase, which keep 13 of 13 members for each mission.

Also for this kind of analysis we can notice a variation of the distribution of the phases. While in the first chart is reported a clear predominance of the **Expiry Timer** phase, whose keeps more than 50% of all the failed calls, in the second one we register a higher distribution of the of phases, in particular for what concerns the **Paging** phase.

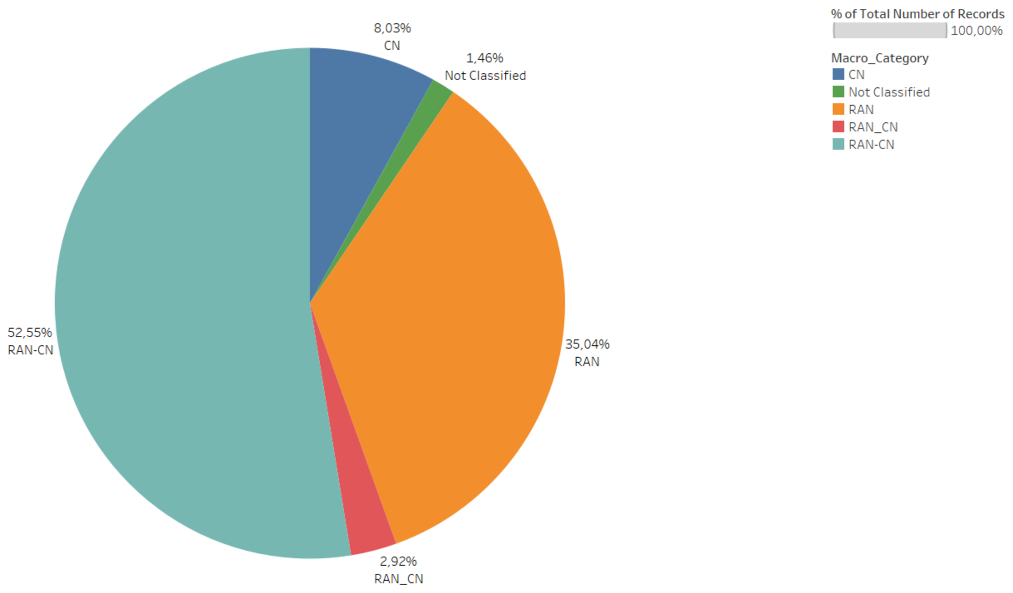


Figure 25. Pie chart of the number of occurrences of each Category registered during the mission – mission 1.

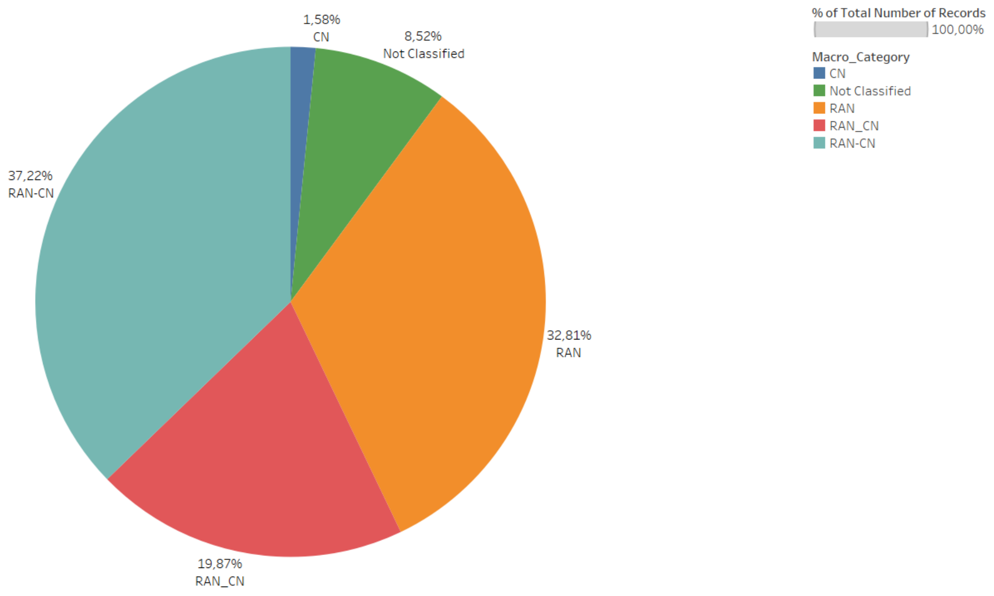


Figure 26. Pie chart of the number of occurrences of each Category registered during the mission – mission 2.

In these two last figures, Figure 25 and Figure 26, is shown the percentage of the total number of records and Macro_Category, represented respectively by size of pie parts and colors. Filters and marks are extrapolated at the same way seen before. Also in this case, for the second mission we can notice a more equal apportionment of the registered categories with respect the first one.

A clear evidence, once analyzing all these charts, is that, even if mission 2 have a smaller duration with respect the other one (mission 2 was only three days, while mission 1 five days), it presents a higher number of failed calls, as demonstrated also in the graphs, 317 versus 274. This is the kind of analysis that can become useful for clients. In general, they notice that something happened between June and October (the month of the missions) and so they can make some analysis in place discovering maybe some new problems.

4 ANOMALY DETECTION

4.1 INTRODUCTION

Once the data have been obtained following the correlation and comparison operations with the analyst outputs shown in the previous chapter, we have tried to apply this information in an innovative context such as that of the anomaly detection. Specifically, anomaly, or outlier, detection is a technique generally used to identify objects, events or observations, which differ significantly from most of the data. There may be several reasons that may cause the presence of an anomaly within a dataset, such as possible errors in measuring instruments or human errors, or simply may represent a suspicious event that differs significantly from a well-defined pattern, as in the case of bank fraud attempts.

Depending on the context in which this technique is used, it must present a certain degree of accuracy and rigidity to avoid possible catastrophic scenarios. This chapter introduces the main characteristics of this complex technique.

4.2 WHAT ARE ANOMALIES?

A possible definition of anomaly was given by from Barnett and Lewis (1994) and says:

> *"An observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data."*

The identification and possible removal of these anomalous observations plays a key role in preserving the performance of a system. Depending on the scope, an anomaly can take a different aspect, can be dictated by a series of factors or simply be the result of a measurement error. The anomaly in an assembly line can be for example a machine that has stopped working, while in a web server it can represent a potential malicious attack to steal sensitive data of a company or a certain clientele for example. The areas of application can be among the most disparate, and this makes it easy to see how the definition of a single pattern for solving this problem is not simple. Real-time production monitoring, image processing for recognition of anomaly objects or design of a credit card usage pattern for fraudulent action are just other context examples where **anomaly detection** is applied. About this last concept, in Figure 27 [8] is shown a pattern for credit card fraud detection and it can be seen as a simple explanation of what is required by anomaly detection systems.

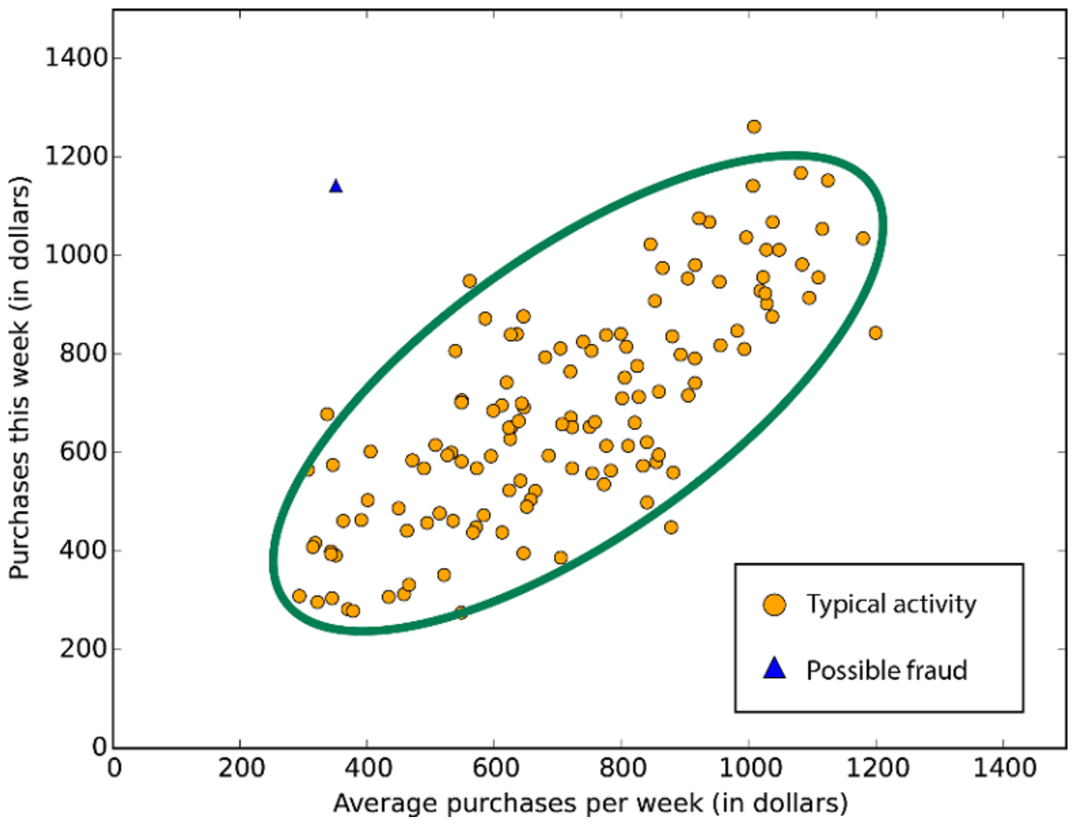


Figure 27. Anomaly detection example of credit card fraud detection.

Among the possible contexts in which the anomaly detection is applied we have fraud detection (fraudulent usage of credit cards), intrusion detection, network performance, fault diagnosis, image processing and activity monitoring (suspicious trends in the stock market for example).

Generally, this technique aims to analyze trends based on time series, or simply on a statistical analysis, similar to what is shown in Figure 27. Once this information has been acknowledged, the system is able to recognize and label a future observation as anomalous or not.

Due to the huge number of possible applications, anomaly detection technique is a collection of dynamic methodologies that can be applied in different ways depending on the application area. The whole system should provide a robust algorithm able to analyze the provided data and, on the top of that, model inliers and outliers points.

4.3 Challenges

The patterns used by the different anomaly detection algorithms aim at identifying a region or several regions within which the elements present in the dataset are included. The region (or regions) where most of the points fall will be classified as inlier, as well as the points within it, while the remaining points outside it as outliers. An example of what has just been said is depicted in Figure 27. The yellow dots are part of an area or region that represents the distribution considered normal of the dataset, while the blue triangle represents a potential as far from the distribution region considered normal. Obviously, the accuracy of the area of the affected region is essential. However, it is often difficult to achieve a good level of accuracy, due to the structure of the data itself. For this reason, very often a preliminary processing operation must be carried out within the dataset. This pre-processing can take place in different ways and will be discussed in the next chapter.

The continuous evolution of the processes under examination and of the patterns also requires constant and difficult updating in the definition of these regions of normality as they must adapt to the new normal behavior. In the case of the recognition of malicious attacks to an information system, for example, the person who performs the action is very often able to disguise his work with that of a normal process, making its identification even more difficult.

Definition of a behaviour as normal or abnormal is not so easy to determine, this because in many cases it can evolve continuously or at least must be contextualized. An example could be a person whose monthly expenses generally do not exceed a certain threshold and which during the Christmas holidays exceeds that threshold of two or three times.

Moreover, the definitions of anomalies, as previously mentioned, vary considerably depending on the context of application. In healthcare, for example, a slight variation of the haematocrit (PVC), which is the relationship between the corpuscular and liquid part of the blood, could cause a kidney disease and considered in this case as an anomaly, but the same variation in a different context, is not necessary an anomaly.

Due to the above challenges, the anomaly detection problem is not easy to solve. Actually, most of the existing anomaly detection techniques are not able to solve a such variety of problems. There are many factors that must be considered during the problem formulation, like the nature of the data, types of anomalies that should be discovered etc.

4.4 TYPE OF ANOMALY

A very crucial aspect that must be considered is the nature of the desired anomaly. For this reason, anomalies can be classified in three different categories [9]:

1. *Point Anomalies*: this is the simplest type of anomaly, where the algorithm looks for a **point anomaly**, individual data instance that differ strongly from the other points. This category can be exactly linked with Figure 27, where the small blue triangle lies outside the boundary of the normal region, and hence it is a point anomaly since it is different from normal data points.
2. *Contextual Anomalies*: in this case we are more interested in finding the anomaly in a specific context rather than a simple point. As we have already seen, the context strongly influences the general resolution of the problem. For this reason, two acts were defined that could serve a sort of standardization of the operation. The first of these attributes takes the name of *contextual attribute*, and serves to define the context of the problem, as might be latitude and longitude in a spatial dataset for example. The second attribute takes the name of *behavioural attribute* and indicates those non-contextual elements of the dataset (such as the average amount of air pollution in the world).

   These attributes help to find out the anomaly behaviour inside the dataset. Specifically, the anomalous behaviour is determined using the values for the behavioural attributes within a specific context.

   An observation can be considered unusual in a certain context, and so catalogued as an anomaly, while it could not be considered as unusual in another one.

Figure 28 [10] report a clear example of this concept.

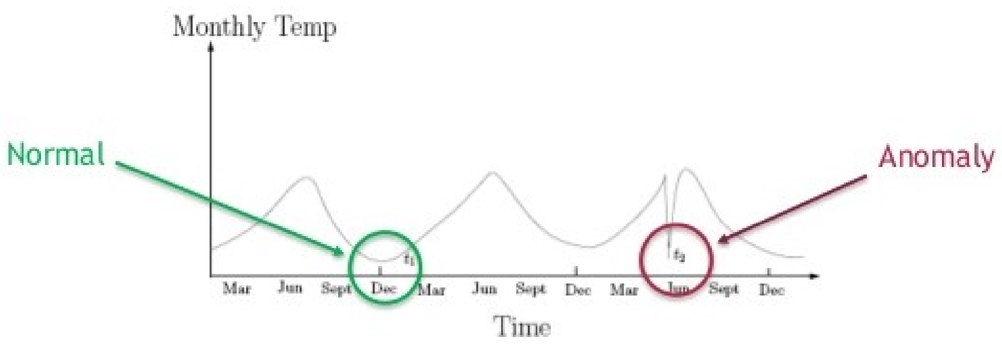


Figure 28. Contextual anomaly detection.

It shows the mean air temperature variation in a Russian city during the year. In this example, point t_2 is considered an anomaly, because it differs from periodic context, even if it takes the same value of a sample that comes first in time, like in t_1.

3. *Collective Anomalies:* a **collective anomaly** is a collection of related data instances considered as anomalous. In this case is required a relationship among data. In Figure 29 [9] is shown an EKG (human electrocardiogram) that can easily explain this concept.

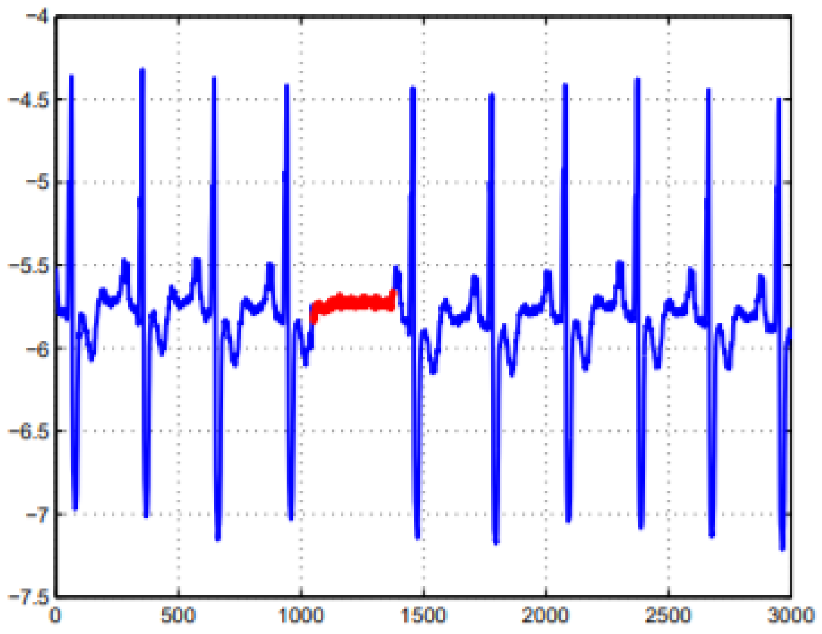


Figure 29. Collective anomaly detection example.

The red part highlights the points that should not be considered anomalous from the quantitative point of view, as they take on previously displayed values, but their grouping certainly represents an anomaly since it does not follow the general trend of the registered samples, in this case the heartbeat of a human.

4.5 Anomaly Detection Technique

In literature [9] there are three different techniques that can be used:

I. **Unsupervised anomaly detection:** This first technique does not make any initial consideration on the dataset available, based on conviction that the points to be classified as anomalous present a distribution different from all others. With this approach a pattern is then created that aims to define one or more regions of normality based on the concept of density, that is how the points are distributed in space, starting from the portion of data made available.

   Usually, when there is a set of *N* cases, these are divided into two subsets, the first one is used for training the machine, the second one is used to test the machine and verify how well it learned by the first subset. Correct dimension of data train set is fundamental and sometimes is not so easy. This makes this technique widely applicable, because it does not require training data.
   The main limitation of this technique is the fact that not always the assumption that the points to be considered as anomalous present a distribution different from the normal ones. The visual representation of this approach can be found in Figure 30 [11].

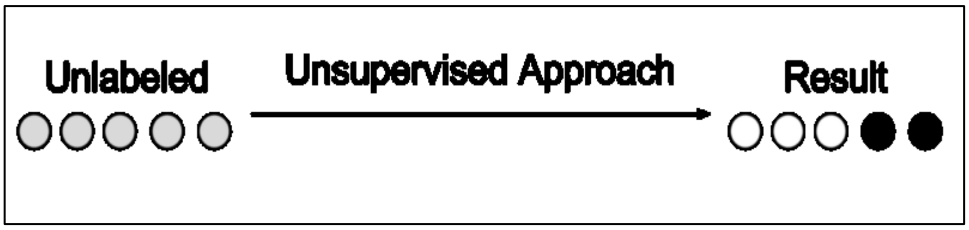


Figure 30. Unsupervised anomaly detection

II. **Supervised anomaly detection:** This second technique can be seen as the opposite to the first one. Here a first classification of data such as inliers or outliers is required. The system, therefore, during the training phase will learn directly from this binary classification and the characteristics of the data themselves. The accuracy of the training phase can then be tested on another portion of the dataset available. In Figure 31 [11] it is possible to view a practical example of this technique. Regions N1 and N2 contain data classified as normal, while outliers are obviously outside these areas. As can be seen, they can be isolated points, as in the case of O1 and O2, or they can also outline an area of abnormality, as for O3.

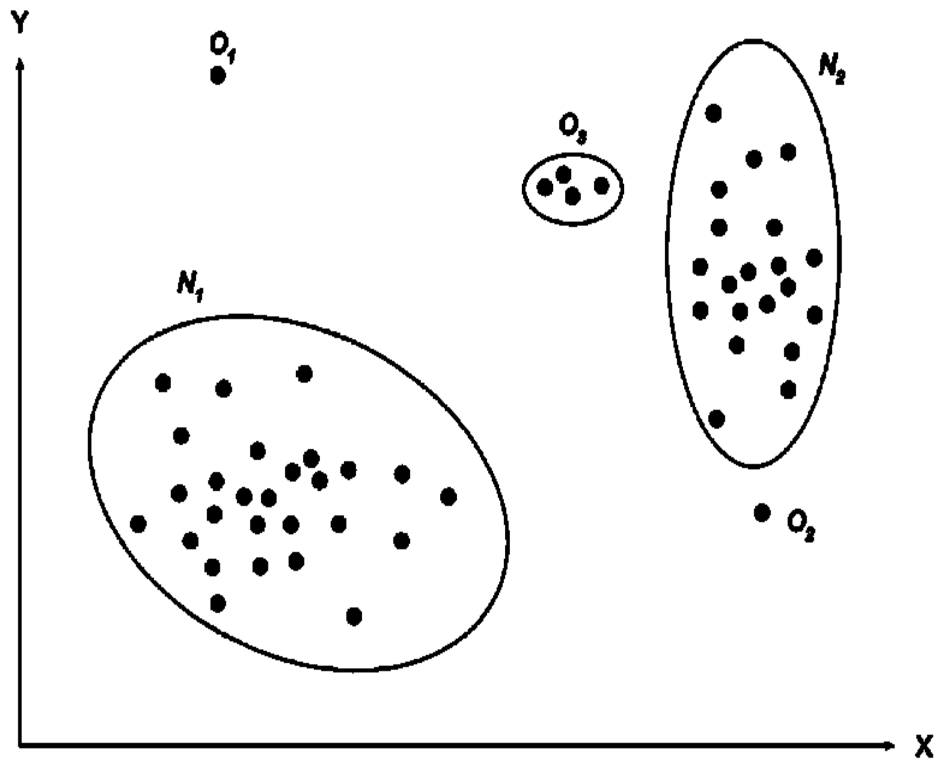


Figure 31. Illustration of simple anomalies in two-dimensional space.

Through this approach a model is created that compares new data with those used for the training phase. Once this appearance is made, the new points are ranked based on what the system learned during the learning process. At the end of this classification, is possible to can compare the result of the model with the real classification of the new points (all points must be classified at the beginning as normal or abnormal). This is probably a more accurate technique than the first, but the limitation of the primary classification makes it of little use, since this operation is not always easy to carry out.

In Figure 32 [11] is illustrated the structure of this approach.

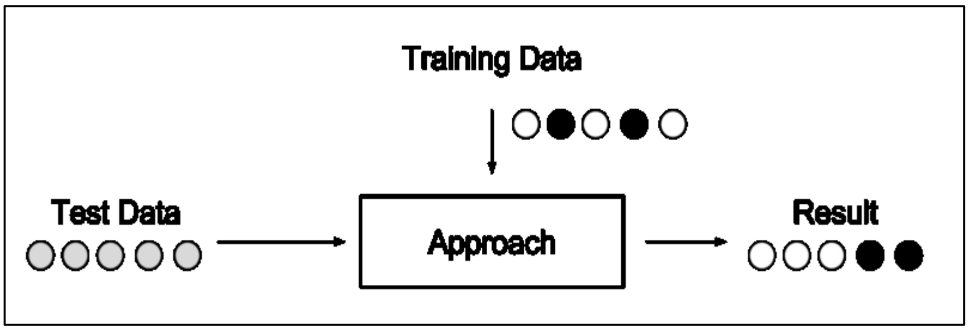


Figure 32. Supervised anomaly detection structure.

III. **Semi-supervised anomaly detection**: The latter technique starts from the consideration that the data available for the training phase are all classified as normal. The model that is created learns the characteristics only of normal data, only one class then. When the test data is provided, the model will be able to classify the new points based on the normal class it has learned. In Figure 33 [11] is illustrated the structure of the semi-supervised approach.

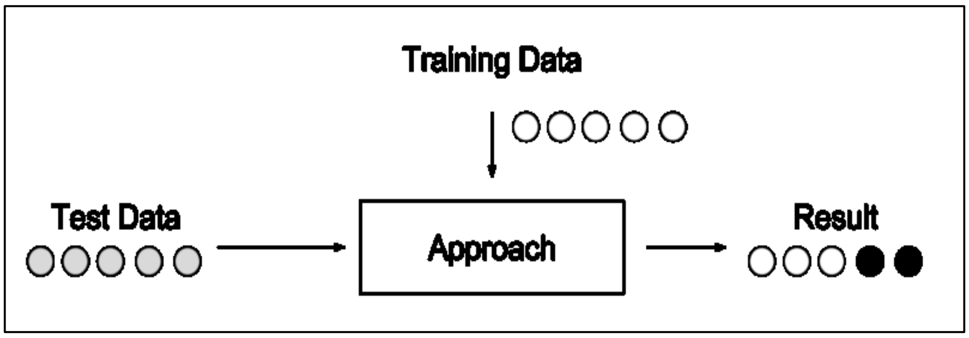


Figure 33. Semi-supervised anomaly detection structure.

4.6 OUTPUT OF ANOMALY DETECTION

Another fundamental point regarding the anomaly detection techniques just analyzed is that of the type of output that is produced. Generally, the output types can be two:

- **Scores**: at each point a percentage is assigned that indicates the probability of being an outlier.
- **Label**s: Each point is labeled as normal or abnormal.

The type of output desired, of course, depends on the need and the objective of the analysis being conducted. Generally, the scores can give a clearer idea than a binary classification.

5 APPLICATION AND RESULTS

5.1 INTRODUCTION

In the previous chapter a complete analysis was carried out on what are the main characteristics of the anomaly detection technique. The following chapter analyzes the results obtained from the application of this technique on the dataset obtained by the algorithm developed for the automation of the main causes of failed calls during the benchmarking missions. In particular, the type of analysis carried out will be described, which envisages a comparison between two missions done in the same area but in different periods of the year, the pre-processing of the data, the algorithms used and obviously the results obtained from the application of these last and final considerations.

5.2 PROBLEM DEFINITION AND DATASET EXTRAPOLATION

Once the data have been obtained with the relevant main causes selected, it was decided to integrate anomaly detection algorithms in such a way as to provide a further analysis tool to the analysts in conducting their analyzes in the fastest and most optimal possible way. To do this, further interviews were necessary in order to direct the work towards requests that were explicitly made to us by analysts. The most logical application of this type of technique, such as those of the anomaly detection, would have been to carry out all the analyzes on a dataset that was designed in such a way that the algorithms used were able to recognize anomalies intended as call failures. However, due to the data available, this solution is not feasible. In fact, the data give information relatively only to the failed calls and to those that are the main causes of the latter. There is no information on successful calls and therefore it is not possible to conduct an analysis of this kind. It was therefore necessary to find another approach that would meet the needs of analysts.

The approach envisages a comparison between two missions that took place in the same geographical region but in different periods of the year, similar to what was already done in the third chapter with the graphical display of trends with Tableau. Similarly, also in this case a quantitative analysis is carried out; specifically, they are compared from the point of view of the number of failures recorded in predefined micro zones of the macro area of interest. These micro zones have been defined through a grid that is superimposed on the area where the events of the two missions were recorded. In Figure 34 and 35 we see these events of the two missions, while in Figure 36 and 37 the grid superimposed on the events is reported.

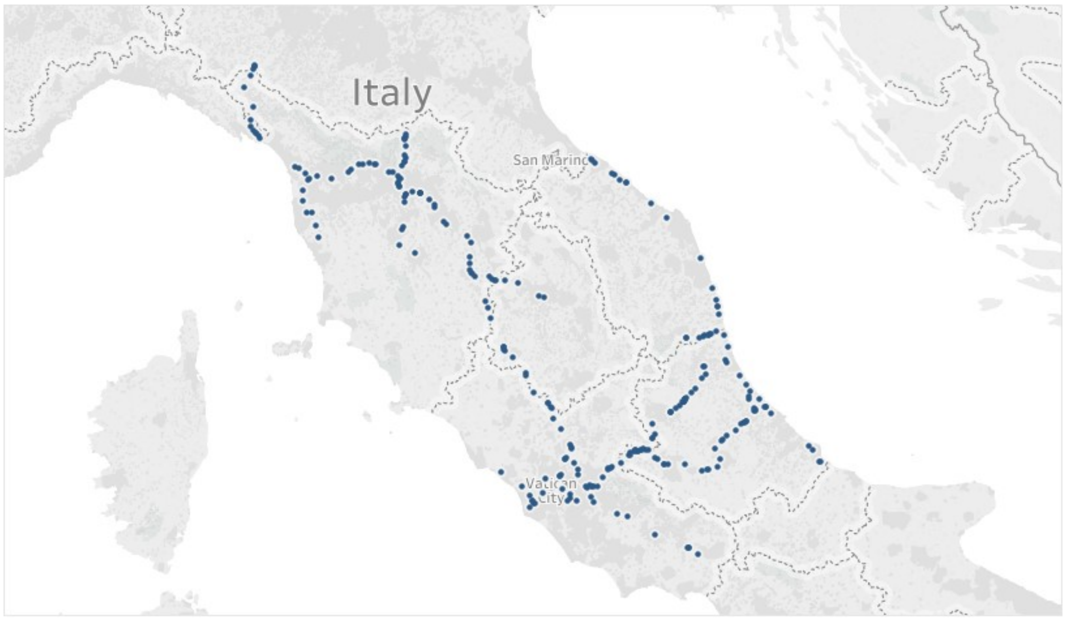


Figure 34. First mission events localization.

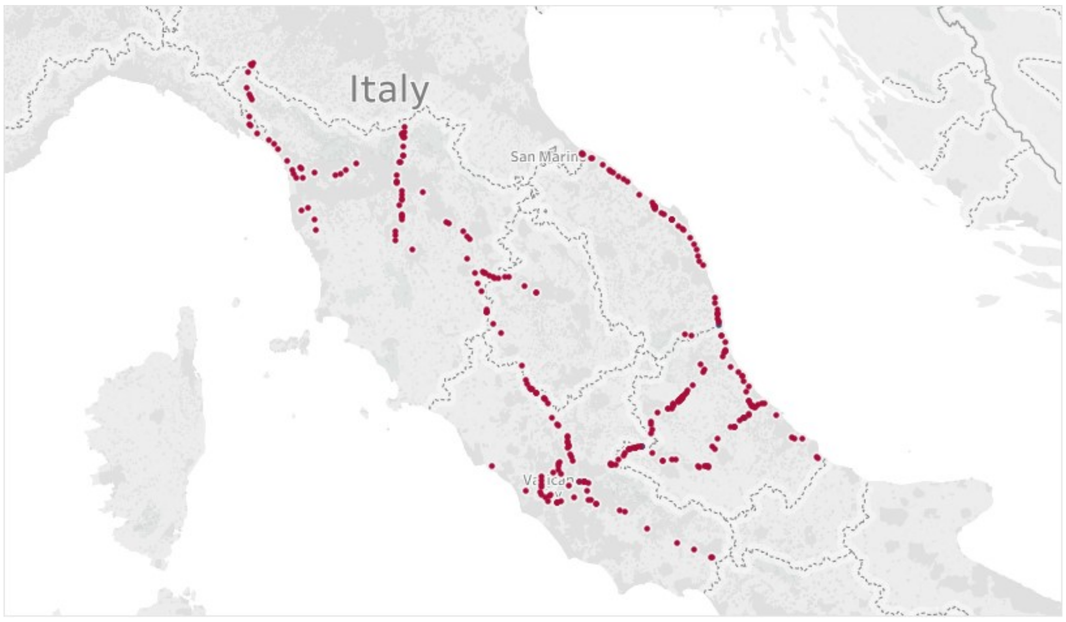


Figure 35. Second mission events localization.

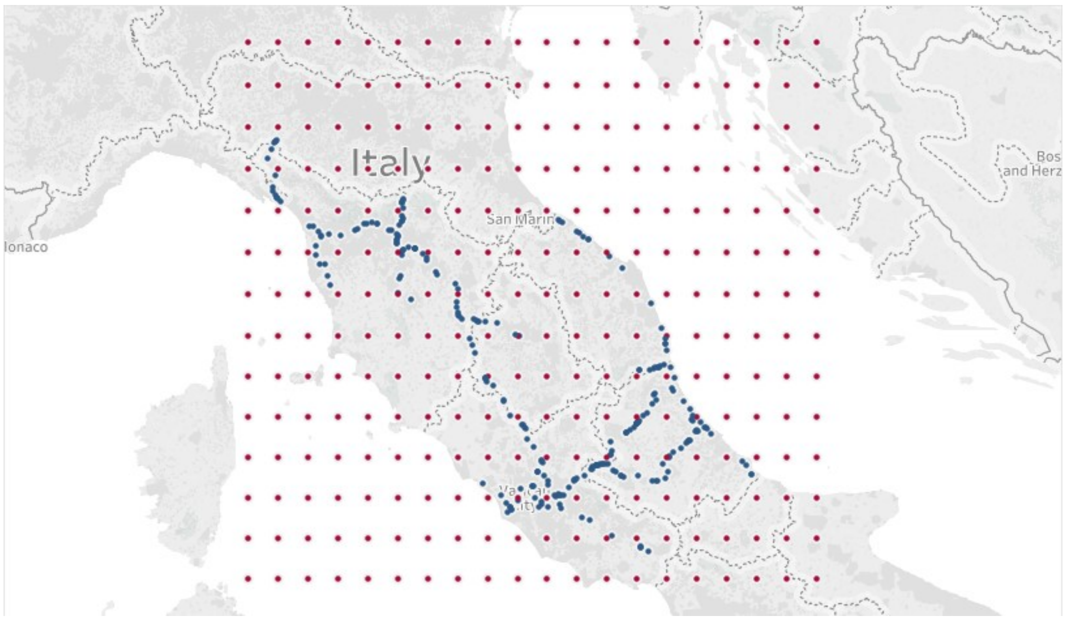


Figure 26. Generated grid over first mission events.

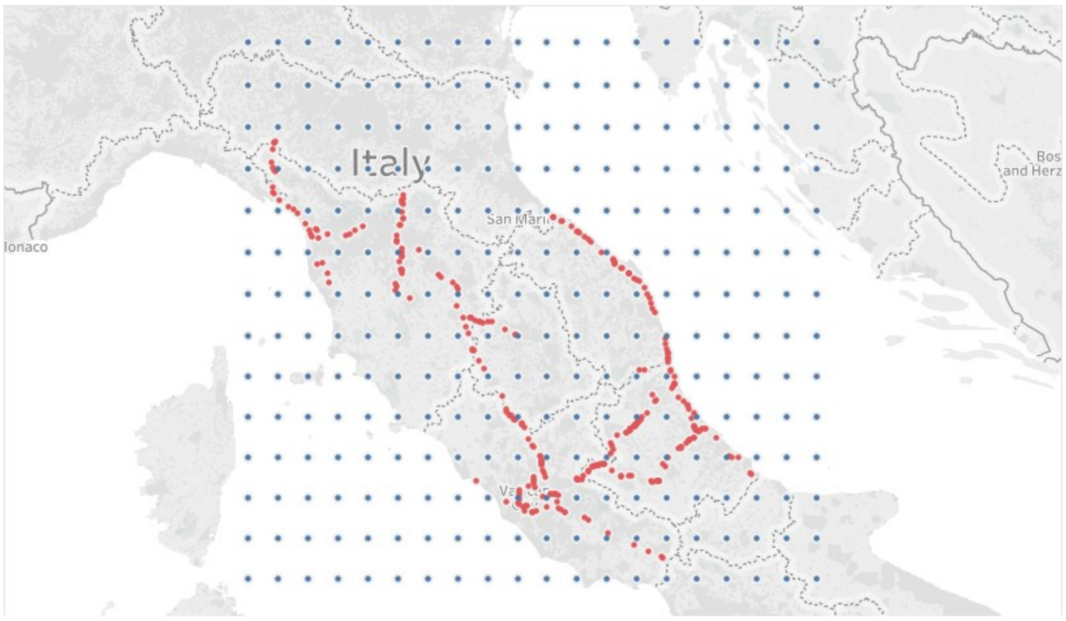


Figure 37. Generated grid over second mission events.

The area of the single micro-zone, also known as pixel, was defined so that it covered an area of about ten square kilometers. At this point, these zones have been labeled with unique identifiers, and subsequently a count of the events that fall within the latter have been made. The comparison that is made, therefore, provides a comparison of the variation in the number of events recorded in the individual areas. In this way, analysts are able to verify the spatial concentration of failed calls and, when an overflow of recorded events should occur, they will be able to conduct a more in-depth analysis of the phenomenon. Analysts are more interested in positive variations, so in seeing if there are actually more bankruptcy events in a given area than the previous mission or not. A possible decrease in the number of events is not in fact an anomaly, since their main task is to provide feedbacks so that mobile operators can optimize their network infrastructure. A decrease in the number of events could mean that the optimization was carried out correctly.

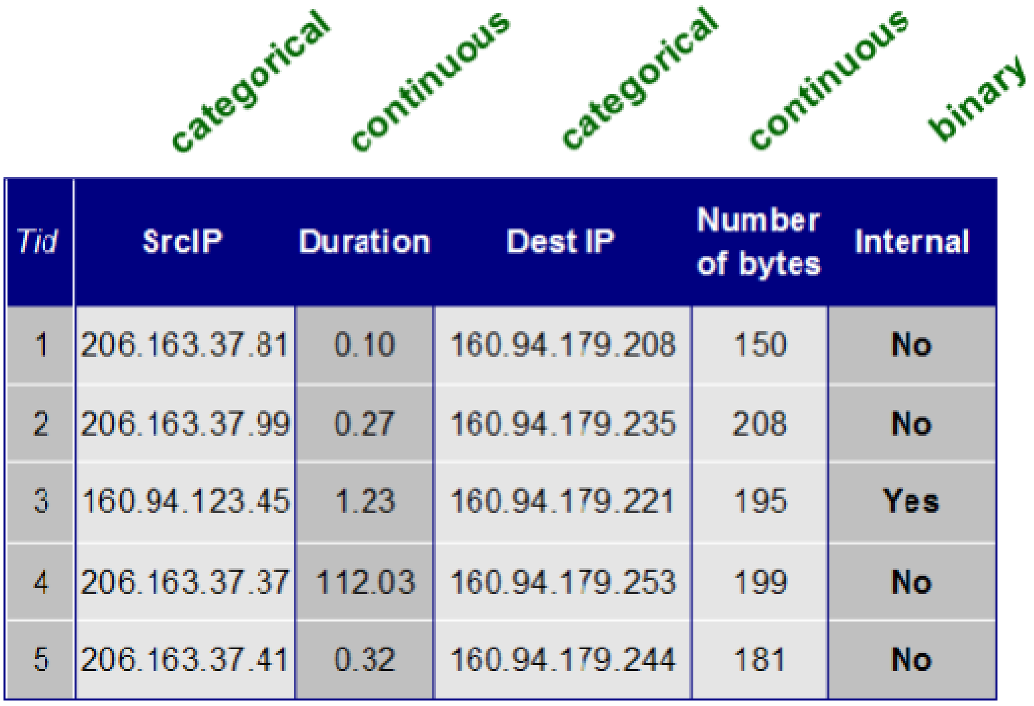
The dataset thus obtained consists of only three features, the first represents the identifier of the micro zone, or pixel, in which at least one event was recorded in one of the two missions, while the other two columns respectively report the number of failures recorded in the first and in the second mission in correspondence of the same micro zone. In Table 3 is shown a little part of the dataset obtained.

Cell ID	**Occurences_First_Mission**	**Occurences_Second_Mission**
1	6	5
2	6	6
3	0	2
4	2	3
5	5	4
6	7	7
7	5	4
8	0	3
9	10	4

Table 3. Dataset example.

One of the fundamental aspects in integrating the technique of anomaly detection lies precisely in the structure and nature of the data that are input [12].
Each given dataset is characterized by the presence of one (univariate) or more attributes (multivariate), where for attributes we generally mean the features that make up the dataset or data instance, which in our case are the identifiers of the micro zones and the number of recorded events. In the case of a multivariate data instance, the correct definition of the data type plays a key role. Generally, the data can be of different types (binary, categorical, continuous etc.). In the multivariate

case, the data can all be of the same nature, or they can all be different, as shown in figure 38.


categorical
continuous
categorical
continuous
binary

Tid	SrcIP	Duration	Dest IP	Number of bytes	Internal
1	206.163.37.81	0.10	160.94.179.208	150	No
2	206.163.37.99	0.27	160.94.179.235	208	No
3	160.94.123.45	1.23	160.94.179.221	195	Yes
4	206.163.37.37	112.03	160.94.179.253	199	No
5	206.163.37.41	0.32	160.94.179.244	181	No

Figure 38. Categories' example of variables.

The data that are supplied to the machine learning algorithms must be presented in a format consistent with the characteristics of the algorithms themselves. From the figure above reported is possible to see how data can be reported in different ways, in the form of numbers, words and so on. Some algorithms such as C4.5, an algorithm proposed by Ross Quinlan in 1994 that allows to efficiently organize a decision tree for classification, can easily deals with nominal/categorical features (it has more problems with numerical features), while other algorithms fail to handle them, such as those used for regression (MSE, Gradient Descent, Steepest Descent, and so on).

As for all machine learning algorithms, the real nature of the attributes that characterize the available dataset determines the possibility or not of applicability of the anomaly detection. For example, for statistical techniques different statistical models have to be used for continuous and categorical data. Similarly, for nearest-neighbor-based techniques, the nature of attributes would determine the distance measure to be used. Often, instead of the actual data, the pair wise distance between instances might be provided in the form of a distance (or similarity) matrix [12].

5.3 Related Work

In the previous chapter the anomaly detection technique was presented in the aspects that most characterize it. Furthermore, the main approaches with which this technique can be used have been presented. Basically, there are two categories of possible approaches: *supervised* and *unsupervised*, with the intermediate variant called *semi-supervised*. The approach to be used depends strongly on the type of data available and on the considerations that are made on them. In particular, the supervised approach requires that the data provided during the training phase have been previously classified as normal or abnormal, so as to create a model that, once input into a new set of data, is able to determine to which class they belong. However, it has already been explained how this type of *a priori* cataloging is not always available for the given data, as this is a complex operation to be carried out and that often takes a long time. To overcome this problem, the unsupervised approach is often preferred. In this case, no primary cataloging is necessary. The main assumption is that the points that should be classified as abnormal present a distribution that is significantly different from that of the normally distributed points. The system based on an approach of this type is therefore able to input unscaled data, just like in our case, and be able to find possible outliers, always based on the belief that most of the data should be cataloged as normal, and only a certain percentage, usually very small, is considered abnormal.

Given the structure of the data obtained and the type of problem that we have set ourselves to solve, this type of approach is ideal. In literature there are many algorithms based on this unsupervised approach. Some of them are shown in the taxonomy in Figure 39 [13], others have also been developed based on new techniques that are not reported here [14, 15, 16].

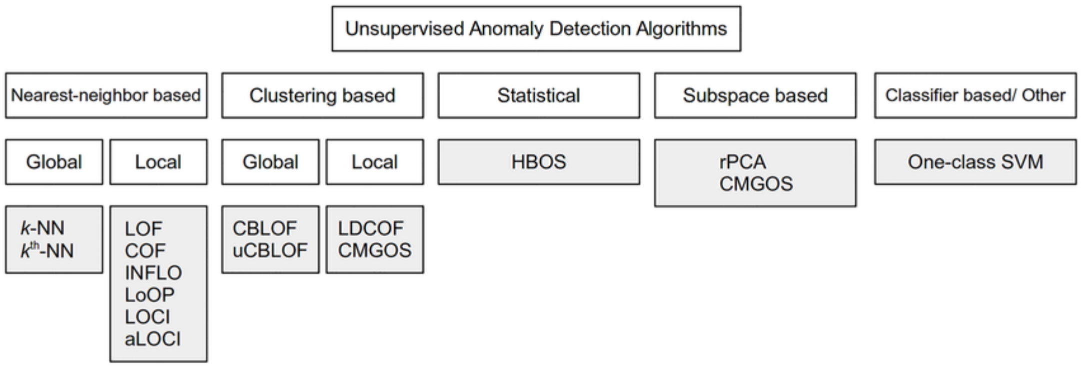


Figure 39. A taxonomy of unsupervised anomaly detection algorithms.

The choice of a specific subset of algorithms to be implemented was not easy and their development from scratch would have required a considerable time. For this reason, it was decided to opt for algorithms developed by scikit-learn [17], a Python module integrating a wide range of state-of-the-art machine learning algorithms for medium-scale supervised and unsupervised problems.

Therefore, below we report a description of the algorithms used and their main ideas.

5.4 CLASSIFIERS COMPARISON

5.4.1 Elliptic Envelope

The distribution of data has been repeatedly placed at the center of the explanations of the anomaly detection approaches. Depending on the type of distribution presented by the data, specific algorithms can be used. In the case in which the data distribution is of the Gaussian type, it is common use to use the Elliptic Envelope.

It is an outlier detection that fits a robust covariance estimate to the data, and thus fits an ellipse to the central data points, ignoring points outside the central mode [18]. This algorithm is based on the minimum covariance determinant method, developed by Rousseeuw in 1984, which is able to recognize the anomalous points whose covariance matrix has the lowest determinant. Thus, first the algorithm fit a minimum covariance determinant model and then it computes the Mahalanobis distance, which gives a measure of the distance between a generic point and the data distribution, in order to evaluate the 'outlyingness' of a point.

5.4.2 Isolation Forest

One of the main limits of the Elliptic Envelope is the high dimensionality of the dataset. In these cases, it is preferable to use a method known as Isolation Forest. This is able to recognize an anomalous point without using conventional measurements based on distance or data density but taking advantage of the concept of isolation. The algorithm randomly selects a point between the minimum and maximum value of the indicated feature. This partitioning process is recursive and can be easily represented through a tree structure. The number of required partitioning to isolate a sample is the path length from the root node to the terminating node. The averaged path length, over a forest of such random trees, is translated as the measure for making the final decision. The shorter the path the more the abnormality [19].

5.4.3 Local Outlier Factor (LOF)

A further method used when the dataset has a high dimensionality is the Local Outlier Factor (LOF). This measures the local density deviation of the individual points to their neighbours. All points with a density that is particularly lower than the average points will be classified as abnormal. At this point a score is developed indicating the level of abnormality of the points analysed, based on the k-nearest-neighbours method.

To calculate the LOF score, there are three steps to evaluate [20]:

- First, the k-nearest-neighbors is developed for each point in the dataset;
- Based on these newly obtained measurements, the local reachability density (LRD) is computed:

$$LRD_k(x) = 1/\left(\frac{\sum_{a\in N_k(x)} d_k(x,o)}{|N_k(x)|}\right) \tag{1}$$

- Finally, the LOF score is developed comparing LRD of the single point with those of its neighbors:

$$LOF(x) = \frac{\sum_{a\in N_k(x)} \frac{LRD_k(o)}{LRD_k(x)}}{|N_k(x)|} \tag{2}$$

In this way the LOF presents itself as a ratio between the local densities of the dataset points. The points to be considered as anomalous, which have a low local density, will have higher scores. This is why this algorithm is called local, because it is interested in the neighbours of the single point.

5.4.4 One Class-SVM

One-Class Support Vector Machine (SVM) is an algorithm introduced for the first time by Schölkopf that is often used in the semi-supervised approach, so when the data that is passed in the training phase to the system do not present anomalous data, or anyway they are present in a small percentage. In this way, a model is created that classifies as anomalous all those points that differ significantly from the data present during the training phase. Nevertheless, it is still used also with unsupervised approaches, through the definition of a soft-margin. In this way, the points analysed are scored by a normalized distance to the determined decision boundary. Traditionally, SVMs algorithms are supervised learning models used for classification able to build a model that assign new examples to one category or the other (there are also different version when possible classification classes are more than two, i.e. OVR or OVO), making it a non-probabilistic binary linear classifier. In contrast to traditional SVMs, the One-Class SVM method uses an implicit transformation function $\varphi(\cdot)$ defined by the kernel to project the data into a higher dimensional space. In this way, it tries to learn a decision boundary (a hyperplane) that achieves the maximum separation between the points and the origin. The points that fall within the decision region will be classified as normal (+1), those outside it as anomalous (-1). The correct definition of the decision boundary is subject to the kernel setting, or Gaussian kernel, which in fact guarantees the existence of this decision boundary. Generally, the most commonly used kernel is the radial basis function (*rbs*). It is applied on two samples x1 and x2, represented as feature vectors, and defined as:

$$K(x, x^{\mathrm{I}}) = exp\left(-\frac{\|x - x^{\mathrm{I}}\|^2}{2\sigma^2}\right) \tag{3}$$

Where $\|x - x^{\mathrm{I}}\|^2$ is the squared Euclidean distance between these two feature vectors; σ should be greater than 0.

It presents itself as a problem of quadratic complexity, whose objective function is as follows:

$$min_{w,\xi i,\rho} \frac{1}{2} \|w\|^2 + \frac{1}{vn} \sum_{i=1}^{n} \xi_i - \rho \tag{4}$$

Subject to:

$$\left(\mathrm{w} \cdot \phi(\mathrm{x}_i)\right) \geq \rho - \xi \qquad for\ all\ i = 1, \dots, n$$

$$\xi_i \geq 0 \qquad for\ all\ i = 1, \dots, n$$

Where ξ_i is the *slack variable*. If this parameter is greater than 1, then x_i lies in the wrong side of the subspace and an error will occur; *n* is the size of the training dataset and *v* is the regularization parameter, also know as the margin of the One-Class SVM, which corresponds to the probability of finding a new, but regular, observation outside the frontier. This parameter, in particular, characterizes the final solution by setting an upper bound on the fraction of the outliers and a lower bound on the number of samples used by the system during the training phase.

By using Lagrange techniques and using a kernel function for the dot-product calculations, the decision function becomes:

$$f(x) = sgn\left(\left(w * \varphi(x_i)\right) - \rho\right) = sgn\left(\sum_{i=1}^{n} \alpha_i K(x, x_i) - \rho)\right) \tag{5}$$

Thus, a hyperplane is formed with distance between the points and the maximized origin, as shown in Figure 40 [21], where the parameter w represents the vector perpendicular to the decision boundary, while ρ is the bias.

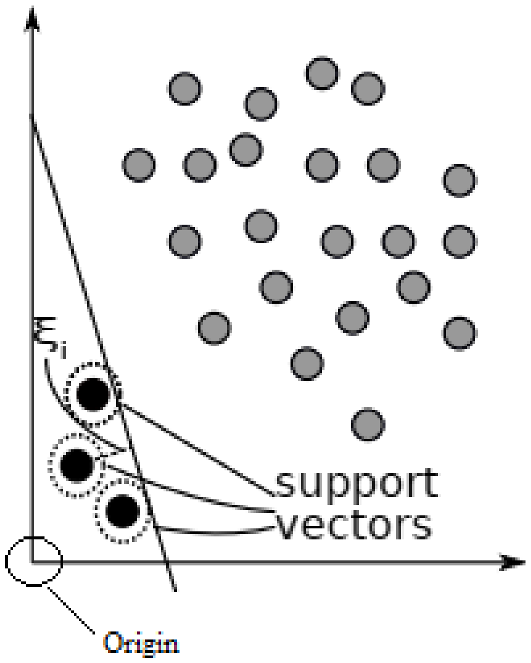


Figure 40. A 2 dimensional example of the decision boundary in the kernel space learned by a one-class SVM.

5.4.5 K-Means

K-means clustering is a simple unsupervised learning algorithm that is used to solve clustering problems. It follows a simple procedure of classifying a given dataset into a number of clusters, defined by the letter "k," which is fixed beforehand. The clusters are then positioned as points and all observations or data points are associated with the nearest cluster, computed, adjusted and then the process starts over using the new adjustments until a desired result is reached.

The assumption is that:

$$y(n) = x(n) + v(n) \tag{6}$$

where x(n) is one among the set $x_1,...,x_K$. Each vector has dimension M. We measured y(n) for $n = 1,\ldots,N$, we don't know the vectors x_k, $k = 1,...,K$ and we have to guess them: *K*-means clustering or hard K-mean algorithm. The algorithm can be summarized as follows:

- we start with an initial guess of $x_k(0)$, $k = 1,...,K$, at the i-th step we associate to $x_k(i)$ the points $y(n)$ which are closer to $x_k(i)$ than to the other points $x_h(i)$ (assignment step or maximization step);
- we evaluate the mean value $m_k(i)$ of the points $y(n)$ that have been associated with $x_k(i)$ (update step or expectation step);
- we define $x_k(i + 1) = m_k(i)$, we set $i = i + 1$, we go back to step 2 until the convergence of the algorithm.

What we obtain at the end of this process is shown in Figure 41 [22], where a comparison between unclustered and clustered data is performed.

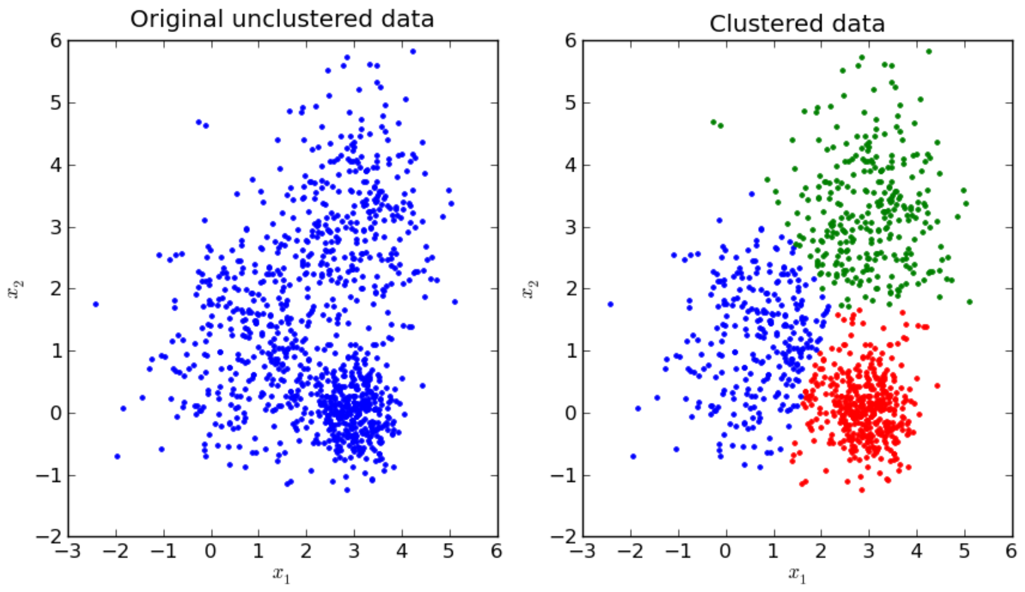


Figure 41. Example of unclustered data with clustered one.

It's important to notice that exists a soft version of the K-means algorithm. This version doesn't make the hard decision in which $y(n)$ belongs to one and only one decision region, but makes a 'soft' decision by associating to $y(n)$ the probability that it belongs to each of the region/cluster $R_1,\ldots,R_k$; then all the N vectors/points y(n) appear in the evaluation of the centroid x_k, but they are weighted by the probability of being in region R_k.

5.5 Experimental Evaluation

5.5.1 Data Selection

The previously obtained data, reported in Table 3, were supplied to a system in which all the previously described algorithms were implemented. In Figure 42 and Figure 43 it is possible to see a graph of the data obtained from the processing that provided the assignment of the failed events in a micro zone or pixel within which they were included.

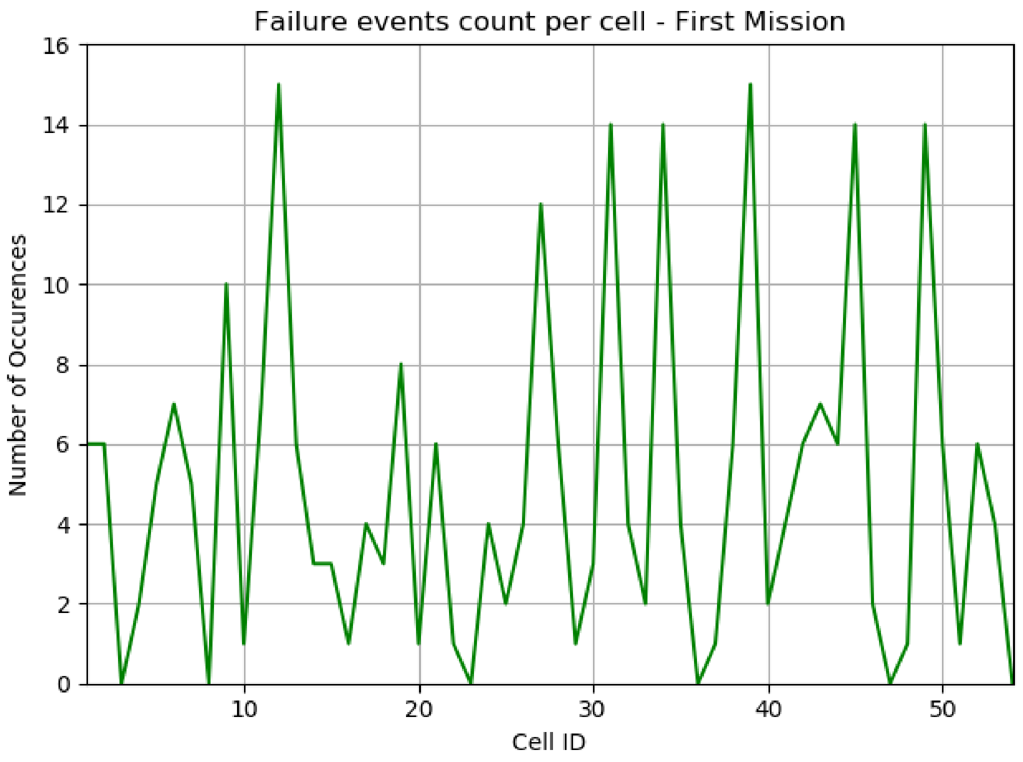


Figure 42. Number of failed events grouped by cells - first mission.

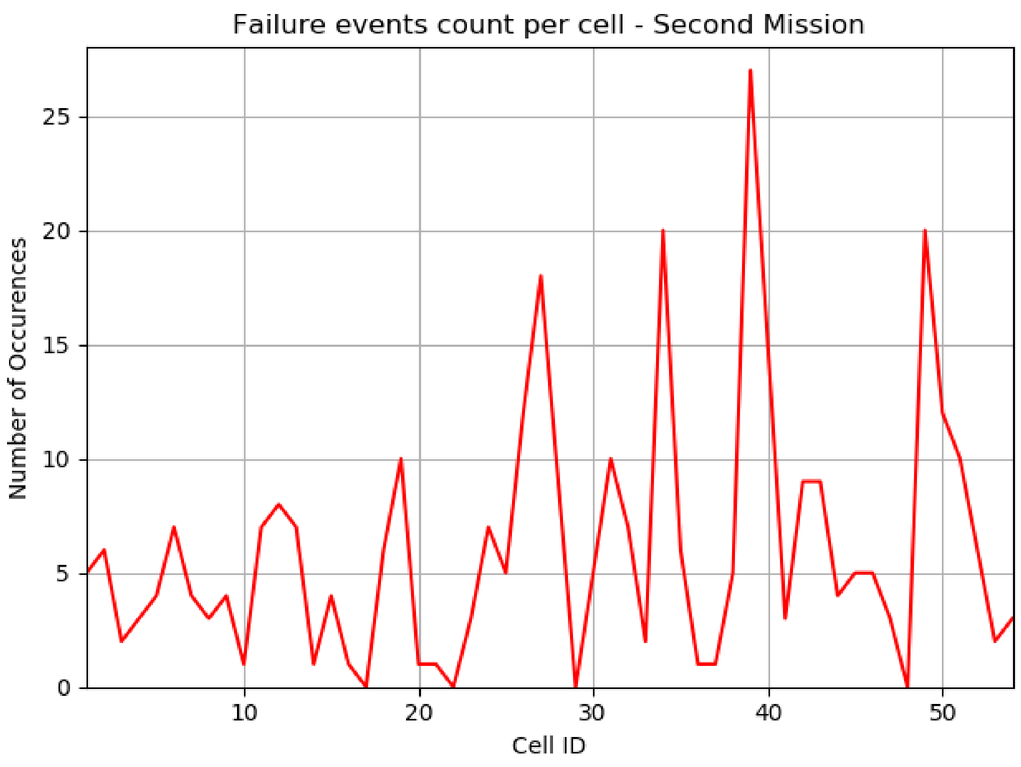


Figure 43. Number of failed events grouped by cells - second mission.

Below, in Figures 44 and 45 are reported the Probability Density Functions (PDFs) of the events of the two missions, through which we define the probability of distribution at each point of the sample space. Similarly, in Figures 46 and 47, the Cumulative Density Function (CDFs), which give information about the distribution of an event before and after a certain point, have also been reported.

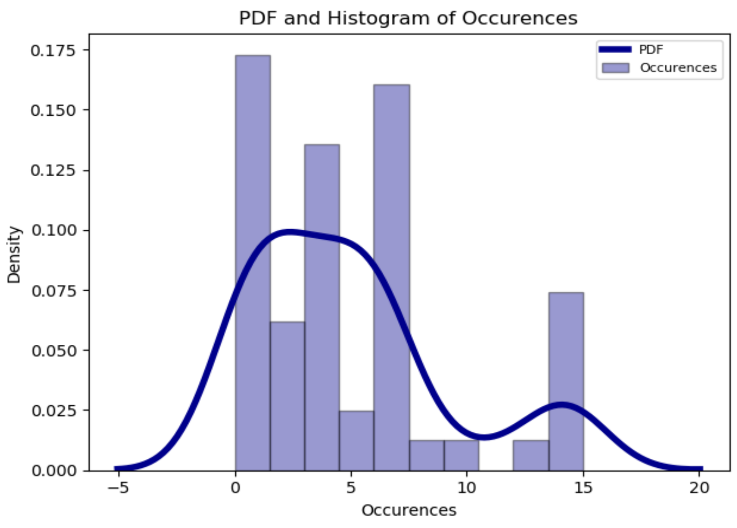


Figure 44. Probability density function of number of failed events grouped by cells - first mission.

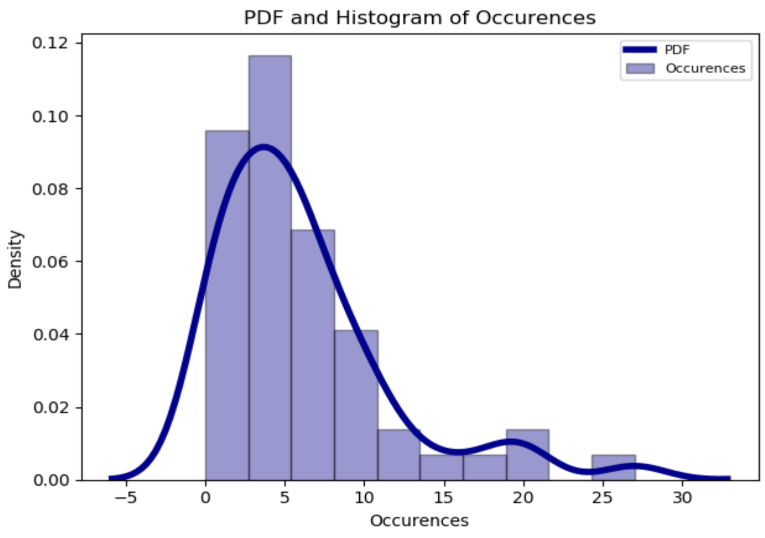


Figure 45. Probability density function of number of failed events grouped by cells - second mission.

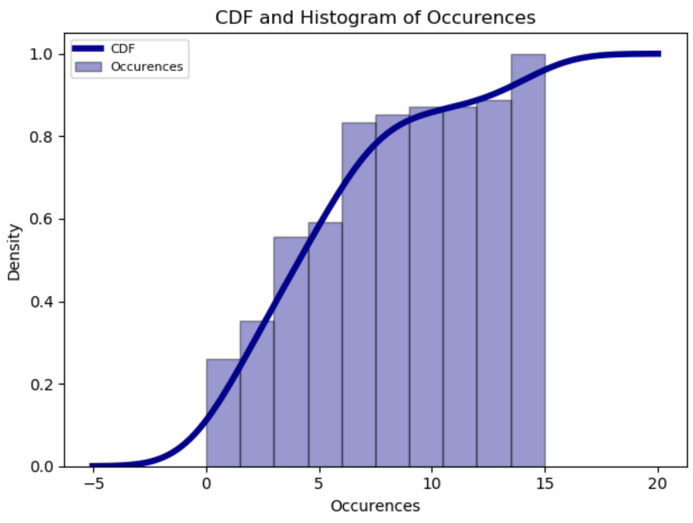


Figure 46. Cumulative density function of number of failed events grouped by cells - first mission.

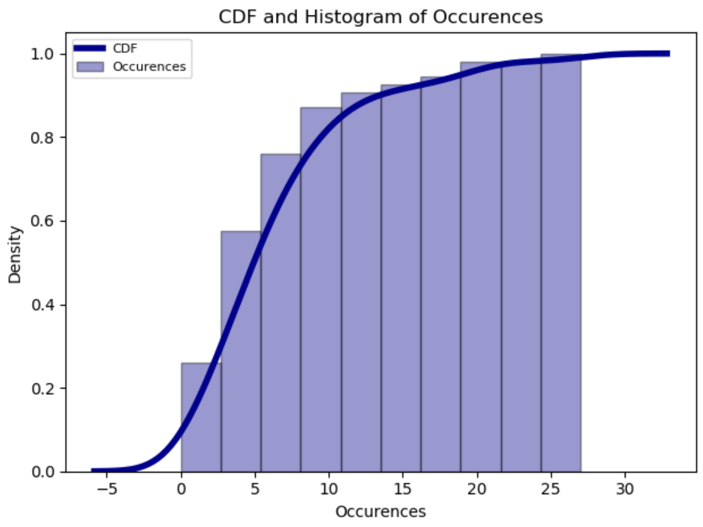


Figure 47. Cumulative density function of number of failed events grouped by cells - second mission.

These are the data with which we fed the system and from which we were able to extrapolate the outliers or points identified as geographical areas in which they were found too many failures. The micro zones, or pixels, so catalogued, were subsequently compared and useful information was extracted to the analysts for a more in-depth analysis of the phenomenon.

5.5.2 Pre-processing

Before being able to carry out the first analyzes, it was necessary to implement a typical operation in the field of machine learning. This operation takes the name of pre-processing and consists in a transformation of the data that are provided to the algorithms, structuring them in this way in a more appropriate and easily understandable way for the systems. The reasons that lead to the use of this technique are many and depend heavily on the structure of the data that is available: often there may be excessive inconsistency or non-representativeness of some features or data may still contain errors or noise. Generally, the same library provided by scikit-learn provides different pre-processing solutions, depending on the needs. In our case, we opted for a method known as MinMaxScaler. With this method, the data are scaled with respect to the maximum and minimum values of the individual columns and are entered in a range between 0 and 1. The following equation shows how the method works:

$$X_{sc} = \frac{X_i - X_{min}}{X_{\max - X_{min}}} \tag{7}$$

5.5.3 Application

Once the data were obtained following the pre-processing, it was possible to start with the training and testing phase. In particular, with the training phase the algorithm is able to establish those that are the main characteristics that are used for classification and create a model on the top of that. Once this phase is over, the testing phase begins, where the accuracy of the classification based on the model created following the training phase is verified. Figure 48 shows a summary of what has just been established.

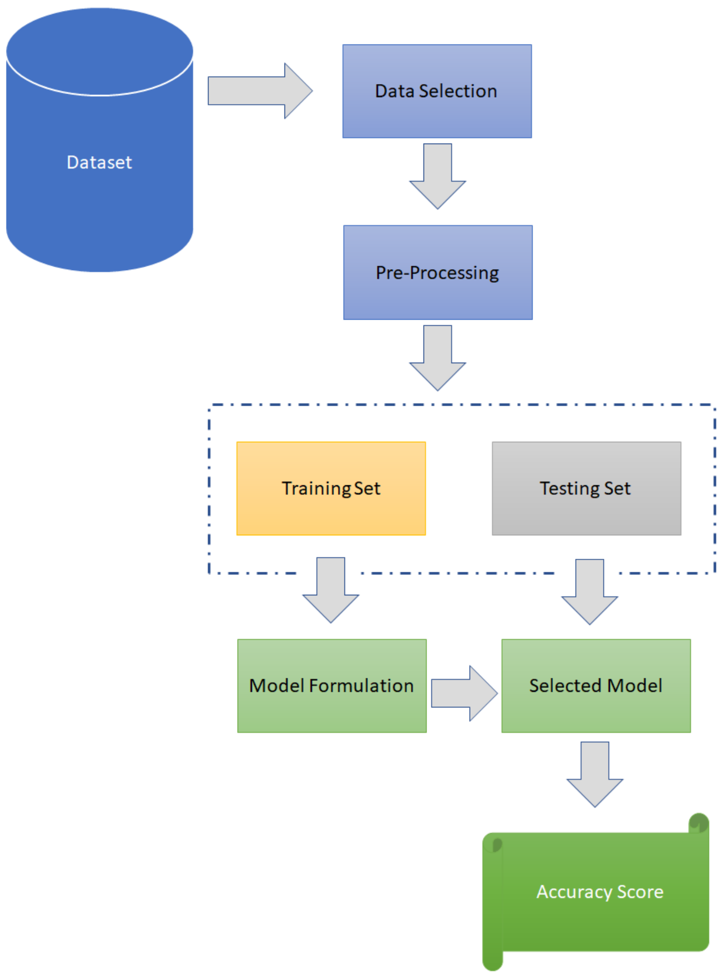


Figure 48. Machine Learning work flow.

From the above image we can see a synthetic representation of the Machine Learning operative process. The data selection and pre-processing of the dataset have already been presented. It is common to divide the available dataset into two subsets, one for training and one for testing. In this case, being the dataset at our disposal too small and being our main goal to find outliers in the entire mission, we decided not to make any division but to use the whole dataset both during training and during testing. In this way, we apply the model on the same portion used for its creation. Thus, we can identify the anomalous points on the entire sample space available. This operation is carried out both for the first and for the second mission. In other words, to analyze the first mission we used the 'Cell ID' and 'Occurences_First_Mission' features, reported in Table 3, both during training and testing. For the second mission, similarly, we used the 'Cell_ID' and 'Occurences_Second_Mission' features. The outliers thus obtained from the two, or the micro zones with excessive values of failures, will be successively compared. This is the final goal of this phase of study.

The following Table 4 shows the configurations of the individual algorithms used for the analysis. The contamination parameter has been set equal to 0.15. It represents the estimated total contamination within the dataset (for example, the proportion of outliers) and is used to define the threshold on the decision function of the algorithms used. The *nu* parameter of the One-Class SVM, instead, represent an upper bound on the fraction of training errors and a lower bound of the fraction of support vectors.

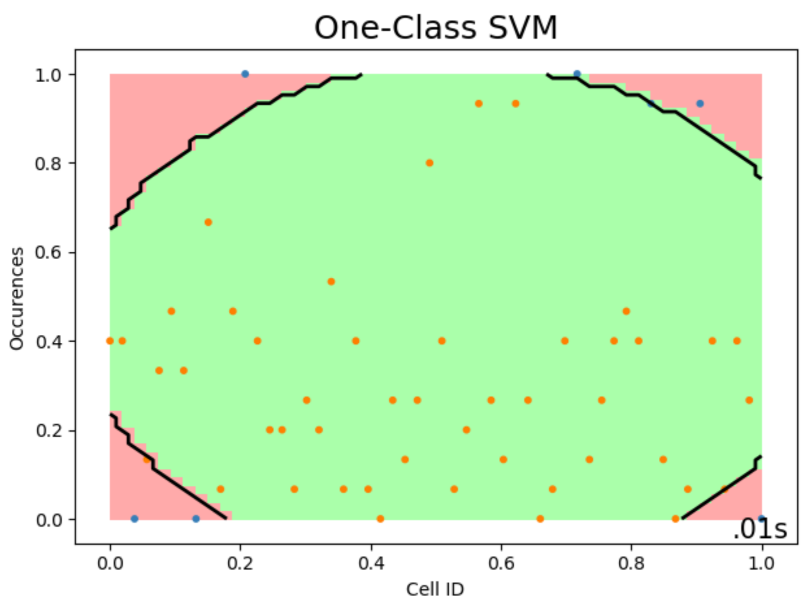
Algorithm	Configuration
K-Means	KMeans(n_clusters=k, max_iter=1000).fit(output.values)
LOF	LocalOutlierFactor(n_neighbors=j, contamination=outliers_fraction).fit_predict(output)
Elliptic Envelope	EllipticEnvelope(contamination=outliers_fraction).fit(output).predict(output)
One-Class SVM	svm.OneClassSVM(nu=outliers_fraction, kernel="rbf", gamma=0.1).fit(output).predict(output)
Isolation Forest	IsolationForest(contamination=outliers_fraction).fit(output).predict(output)

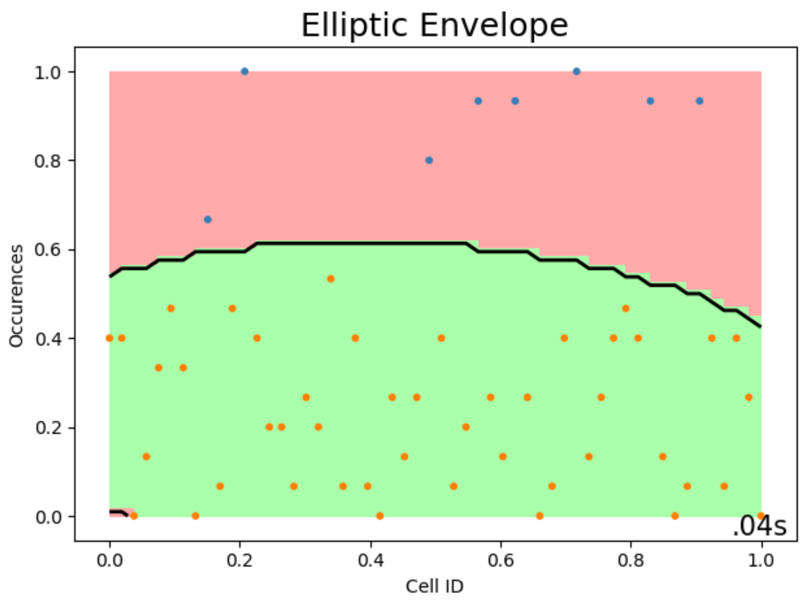
Table 4. Algorithms' configuration

Once the setup of the individual algorithms with the appropriate parameters was completed and the data to be analyzed were supplied to the system, we proceeded to the final analysis of the results obtained. While the configuration of the One-Class SVM, Elliptic Envelope and Isolation Forest methods took place in a static manner, those of the K-Means and LOF were decided following various iterations in which the number of clusters was changed in the case of the first and the number of neighbors for the second. This is because these last two methods are based on the concept of proximity and spatial distribution of data. Therefore, the results shown are those obtained with the best configuration following the performed iterations. To evaluate the goodness of the classification of the algorithms, a manual classification of the same events given as inputs to the system was carried out, and the two results were then compared using different metrics. Therefore, in the next paragraph, different metrics will be analyzed on the basis of this comparison.

5.6 Results and Performances

Figure 49 shows the decision boundaries of the One-Class SVM, Elliptic Envelope and Isolation Forrest for the first mission. These boundaries represent the result of a process that generates a hypersurface where all points are separated into two different sections, one for each class present. In the case of anomaly detection, these boundaries separate the points classified as normal compared to those classified as anomalous. Decision boundaries have not been reported for the Local Outlier Factor (LOF) and the K-Means, because these have no predict method to be applied on new data when they are used for anomaly detection problems.





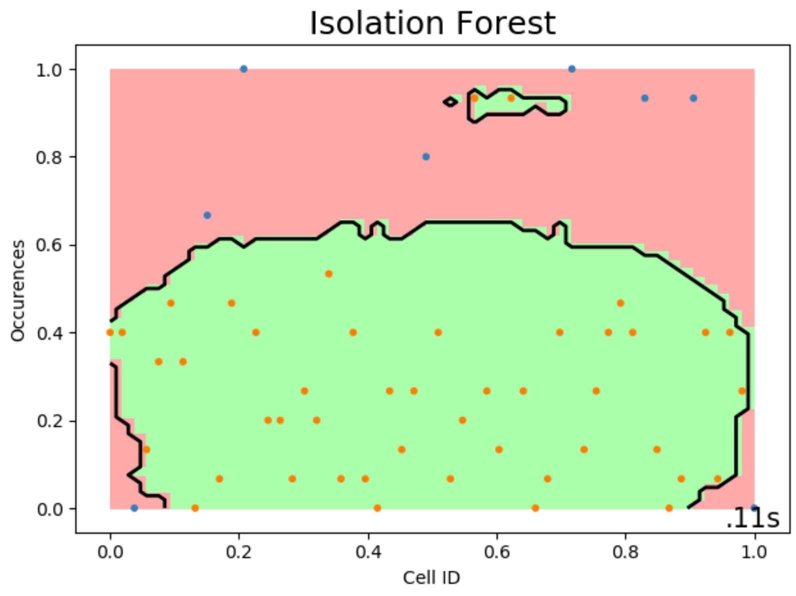
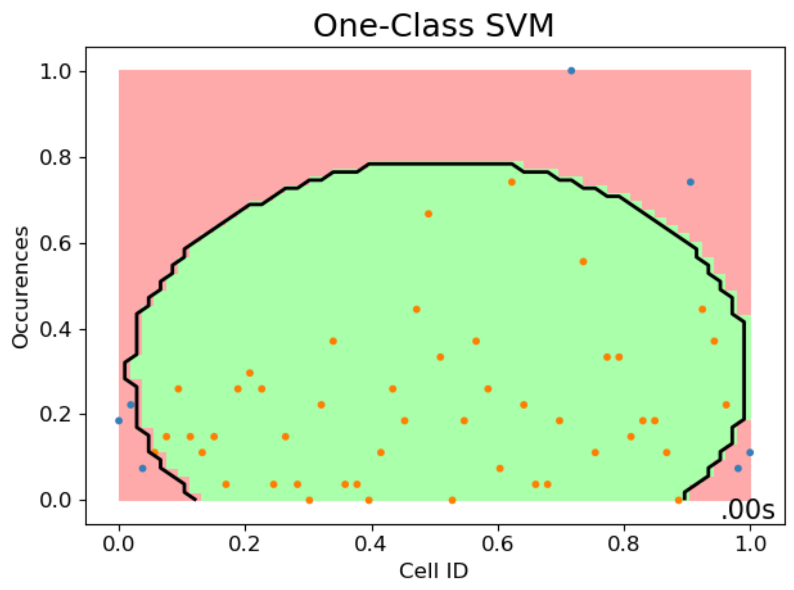
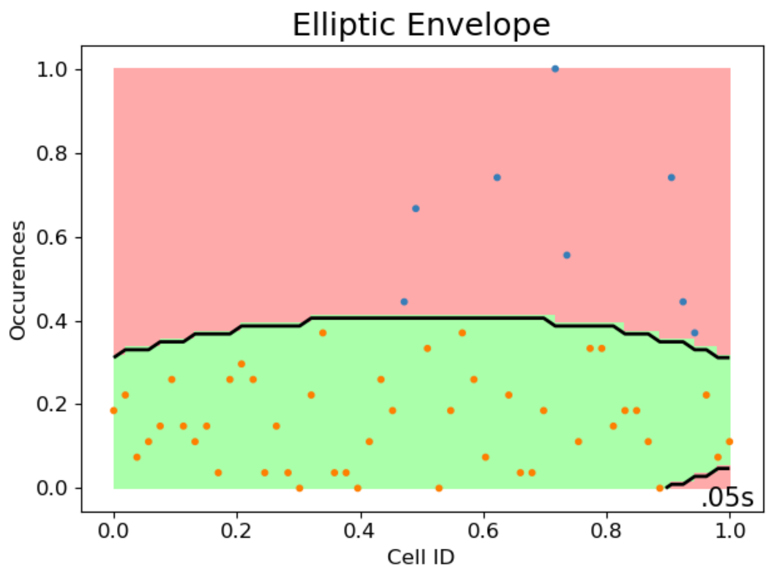


Figure 49. Decision boundaries of One-Class SVM, Elliptic Envelope and Isolation Forrest on first mission dataset.

In Figure 50 are reported the decision boundaries evaluated for the second mission.





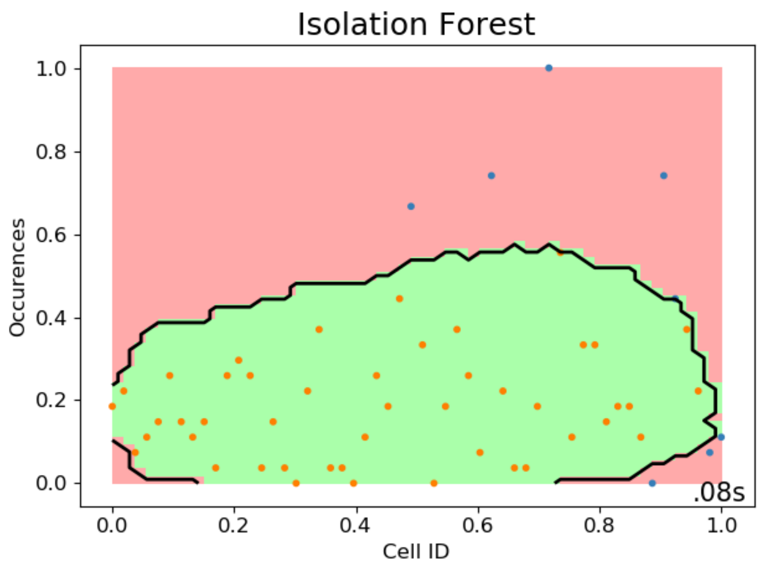


Figure 50. Decision boundaries of One-Class SVM, Elliptic Envelope and Isolation Forrest on second mission dataset.

The points that fall within the green coloured region were classified as normal, conversely those outside this area, then falling within the red zone, were classified as abnormal. As we can see, the individual algorithms have quite different boundaries, also depending on their characteristics previously presented. Figure 51 shows the overlap of the boundaries developed for the two missions.

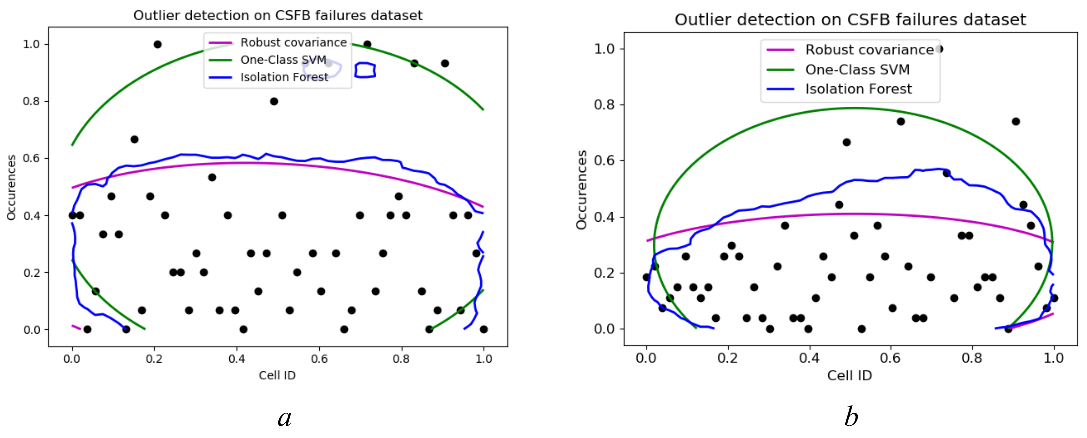


Figure 51. Overlapping decision boundaries evaluated for the first mission (a) and for the second mission (b).

Once the models of the individual algorithms have been correctly defined, it is common practice to develop some metrics in order to evaluate the performance of the models themselves. This is generally not a simple operation as it might seem, since these metrics depend strongly on the type of dataset that is being analyzed and on the degree of imbalance it presents, that is the predominance of one class with respect to the other. For example, if there are two classes, called A and B, it may be sufficient to develop the accuracy of the classifier, understood as the points correctly classified by the specified model. If this accuracy were to reach a fairly high percentage, say 90-95%, then we could conclude that the model operates correctly and that it presents remarkable performances. But in the case of very unbalanced datasets, this consideration is not to be considered more valid as, if we had a class A 90% and class B 10% imbalance, then we would be able to obtain high accuracy only by classifying class A, providing in this case a useless performance metric for the final purpose.

For this reason, there are several metrics that can be considered when assessing the performance of a given model. These metrics are Accuracy, Precision, Recall, F1 score, ROC and AUC. All of these have been developed to draw the final conclusions. They are easily derived from four parameters, so even before going into the details of the individual metrics, it is necessary to give a brief overview of these four parameters and their representation. The so-called Confusion Matrix is shown below, in Table 5. It allows a simplified and intuitive representation of the performances of the model developed, comparing the actual classification of the points present in the dataset, in our case those made by the analysts, and that performed by the model under analysis.

Actual	Positive	True Positive	False Positive
	Negative	False Negative	True Negative
		Positive	Negative
		Predicted	

Table 5. Confusion matrix example.

Table 5 represents an example of the Confusion Matrix. Each column of the matrix represents the instances of the points developed by the model, while the rows show the instances of the points actually classified. From here, four parameters are created. In the table there are the True Positive and True Negative which are highlighted in green. They represent the points that have been correctly classified by the model as positive and negative, where therefore the two classifications coincide. The other two points, False Positive and False Negative, are highlighted in red because it means that the two classifications do not coincide on some points. The goal is to minimize the last two parameters as much as possible. Therefore, we have:

- **True Positive (TP)**: points classified as positive and actually positive.
- **False Positive (FP)**: points classified as positive and actually negative.
- **True Negative (TN)**: points classified as negative and actually negative.
- **False Negative (FN)**: points classified as negative and actually positive.

At this point, we can perform a more in-depth analysis of the metrics used for model performance evaluation. The first that must be considered is Accuracy. It is the simplest and most intuitive of all metrics and represents the relationship between the points correctly classified by the model and all the points. Accuracy is a parameter that is certainly important for the evaluation of performance, but this is valid when there is not a high imbalance between the classes present in the analyzed dataset.

Accuracy is calculated this way:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{8}$$

In addition to the problem of the imbalance of the classes, there is however the fact that the result that provides the Accuracy does not give any insight about the number of False Negatives and False Positives, parameters that are crucial in certain contexts such as the medical and that of bank frauds for example. For this reason, other metrics like Recall and Precision are often preferred.

Recall is defined as the number of true positives divided by the number of true positives plus the number of false negatives.

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

Through the Recall metric we are able to evaluate the ability of a model to find all relevant cases within a dataset. Another very important metric is Precision, understood as the relationship between the TPs and all the positive ones (TP and FP).

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

The main difference between these two metrics is that while Recall expresses the ability to find all relevant instances in a dataset, Precision expresses the proportion of the data points our model says was relevant actually were relevant [23].

Subsequently, there is the F1 score, a metric that is developed as a function of Recall and Precision; in particular, it represents the balance between these two and therefore takes into account both False Positives and False Negatives. Generally, it is more difficult to understand than Accuracy, even if it is more useful, especially if the dataset has an uneven class distribution, as previously discussed.

F1 score is defined as follows:

$$F1 = 2 * \frac{Precision\ *\ Recall}{Precision + Recall} \tag{11}$$

A very important last metric that is taken into account when the performance of a model is developed is the Receiver Operating Characteristic, or ROC, curve. It just shows how the relationship between Recall and Precision changes as also the threshold does. This threshold is a boundary over which the points are considered as positive. By altering the threshold, is possible to achieve the right balance between Precision and Recall. Specifically, this metric shows the relationship between True Positive Rate (TPR) with the False Positive Rate (FPR) as function of the threshold that demarks the boundary between negative and positive classification. These two parameters can be easily obtained from the confusion matrix as follows.

$$TPR = \frac{TP}{TP + FN} \qquad\qquad FPR = \frac{FP}{FP + TN} \tag{12}$$

Generally, we move from left to right decreasing more and more the value of the threshold. By decreasing the value of this threshold, a greater number of points are recognized as positive, thus increasing the True Positives (TP), but there will also be more False Positives (FP). Figure 52 shows an example of what has been stated so far on the ROC curves. The dashed line in black represents a random classifier.

Finally, to obtain even more detailed considerations about the ROC curves of a given model, the total Area Under the Curve (AUC) is often calculated, a parameter between 0 and 1 that represents the area below the ROC curves. The greater the value of this metric, the better the model is.

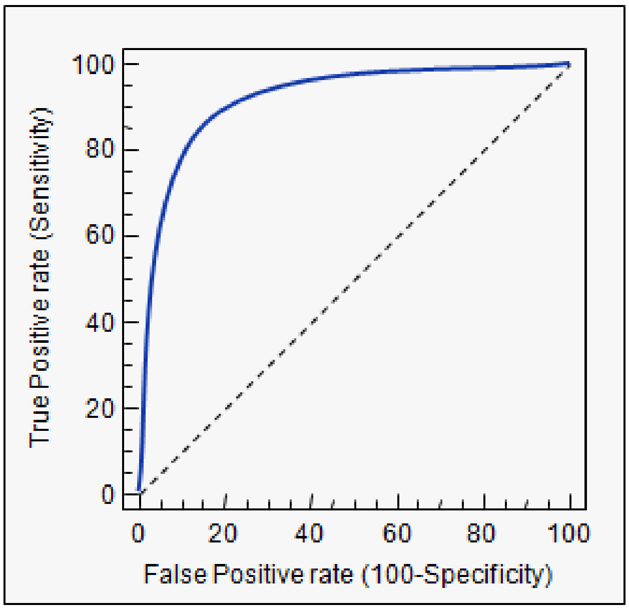


Figure 52. ROC curve example.

Now that a fairly complete overview has been made on the metrics that have been considered to evaluate the performance of the algorithms used for the anomaly detection, we can move on to the actual analysis of the results. Figures 53 and 54 show the Confusion Matrices obtained in the first and second missions respectively from the five different algorithms used.

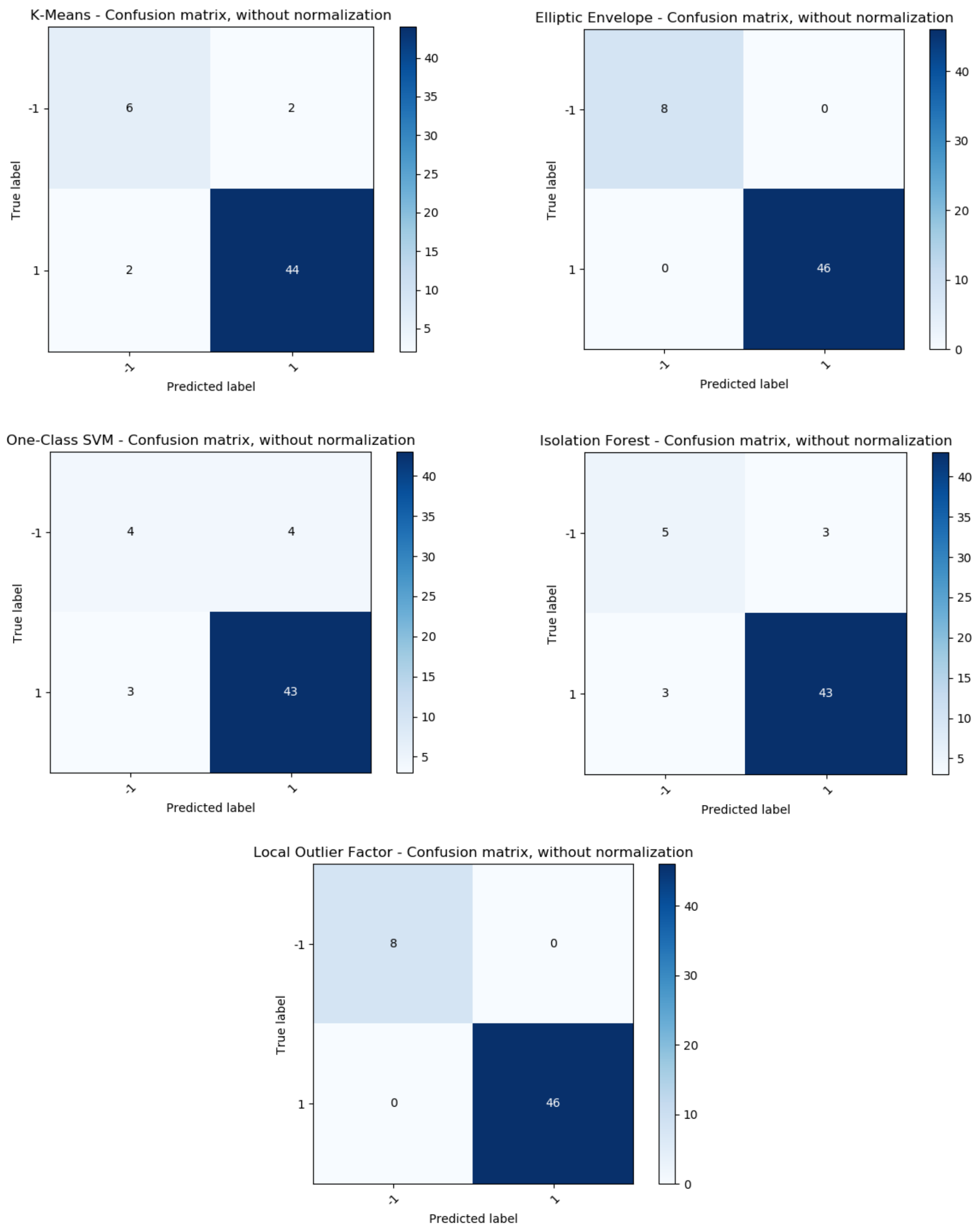


Figure 53. Confusion matrix obtained from the algorithms for the first mission.

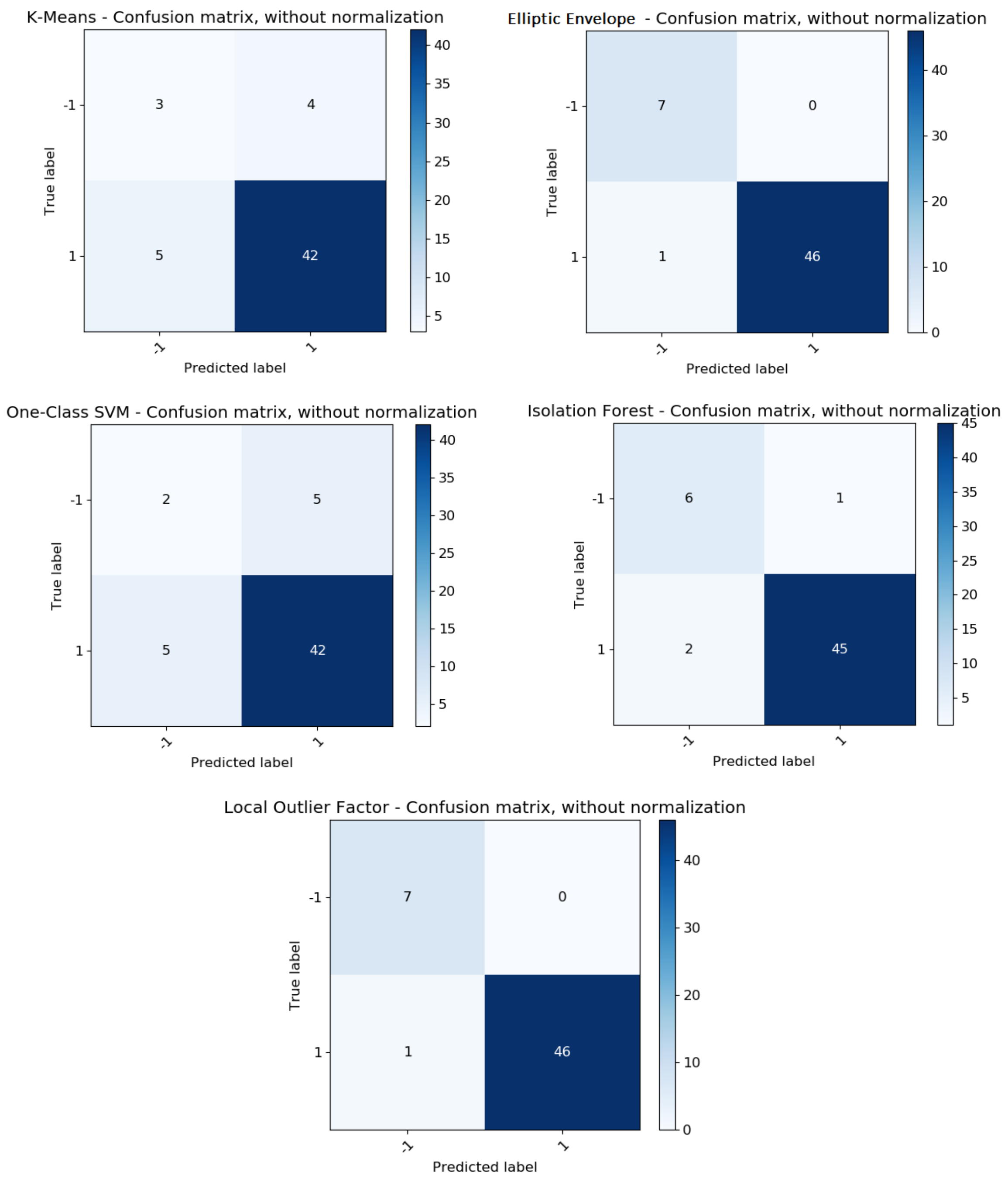


Figure 54. Confusion matrix obtained from the algorithms for the second mission.

In the reported matrices the value 1 represents the positive, -1 the negative. Looking at the individual Confusion Matrices, we notice that both the Elliptic Envelope and the LOF have the lowest number of falses, both positive and negative. This consideration is further confirmed by Tables 6 and 7.

First Mission

Algorithm	Accuracy	Recall	F1	Precision	ROC	AUC
K-Means	0,9259	0,9565	0,9565	0,9565	0,8532	0,8533
LOF	1	1	1	1	1	1
Elliptic Envelope	1	1	1	1	1	1
One-Class SVM	0,8703	0,9347	0,9247	0,9148	0,7173	0,7989
Isolation Forest	0,8888	0,9347	0,9347	0,9347	0,7799	0,9484

Table 6. Metric's results for the first mission.

Second Mission

Algorithm	Accuracy	Recall	F1	Precision	ROC	AUC
K-Means	0,8333	0,8936	0,9032	0,913	0,661	0,6611
LOF	0,9814	0,9787	0,9892	1	0,9893	0.9894
Elliptic Envelope	0,9814	0,9787	0,9892	1	0,9893	1
One-Class SVM	0,8148	0,8936	0,8936	0,8936	0,5896	0,6991
Isolation Forest	0,9444	0,9574	0,9677	0,9782	0,9072	0,9787

Table 7. Metric's results for the second mission.

Regarding the first mission, there is a clear predominance of both LOF and Elliptic Envelope, while in the second mission we observe how there is a slight lowering of performances, which however are not worrying for the final outcome of our considerations, which they see these two algorithms as the best for performances obtained, both in the first and in the second mission. It is necessary to consider, however, that the setup of the Elliptic Envelope has been carried out in a static way, and this makes us understand how the logic that underlies the latter is suitable for a problem of this type, even if the points distribution is not of perfectly Gaussian. The configuration of the LOF, instead, was defined following a series of iterations where the number of neighbors was changed. In the end the configuration was chosen that allowed to obtain the best results, following the analysis of the metrics mentioned previously. In other words, the metrices necessary for performance analysis were developed for each iteration and at the end the one with the best metrics was chosen. Although the various tests performed always result optimal a number of neighbors between 14 and 17, this is not a setting that should be considered correct in an absolute sense, as the dataset that is analyzed always varies and above all the configuration is obtained following a comparison with the classification made by analysts. The latter, as has already been said, is not always available, rather a study has been conducted on the use of unsupervised algorithms for this reason. The manual classification was made in this case only to develop the mentioned metrics. What we are looking for is an algorithm that is able to obtain satisfactory results almost automatically, and the Elliptic Envelope, following the results obtained, is certainly the best possible model for the type of problem proposed. Figures 55 and 56 show the ROC curves obtained for both missions. The curves drawn and the AUC reported in the legend confirm what was previously concluded about the victory of the Elliptic Envelope (Robust Covariance).

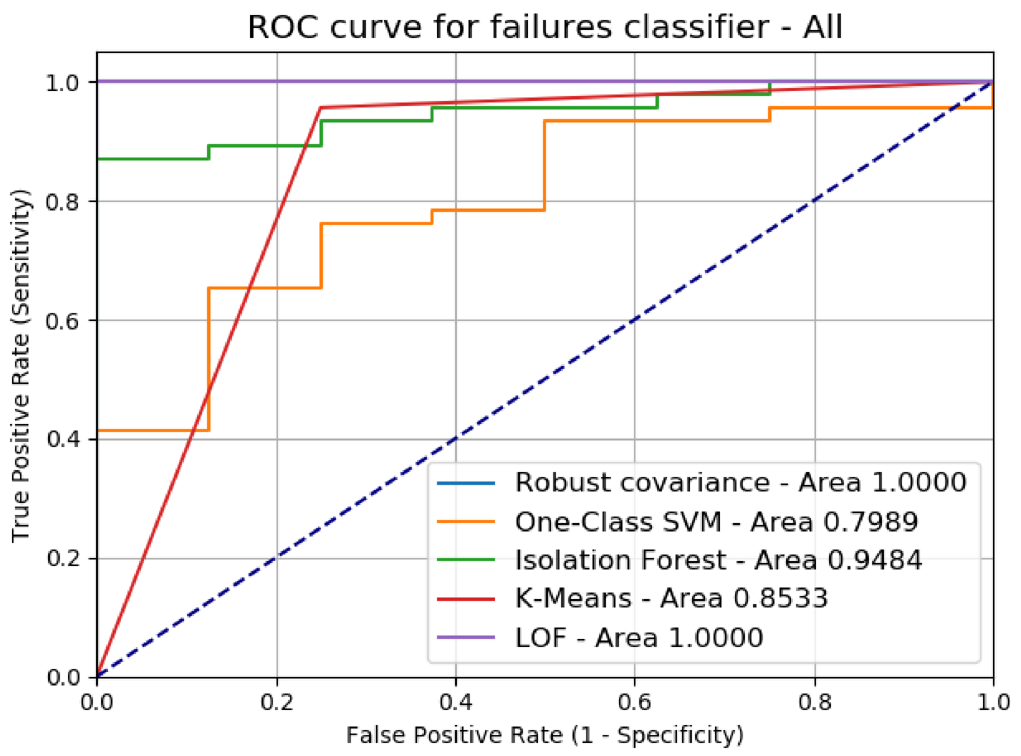


Figure 55. ROC curve and AUC of the algorithm's results for the first mission.

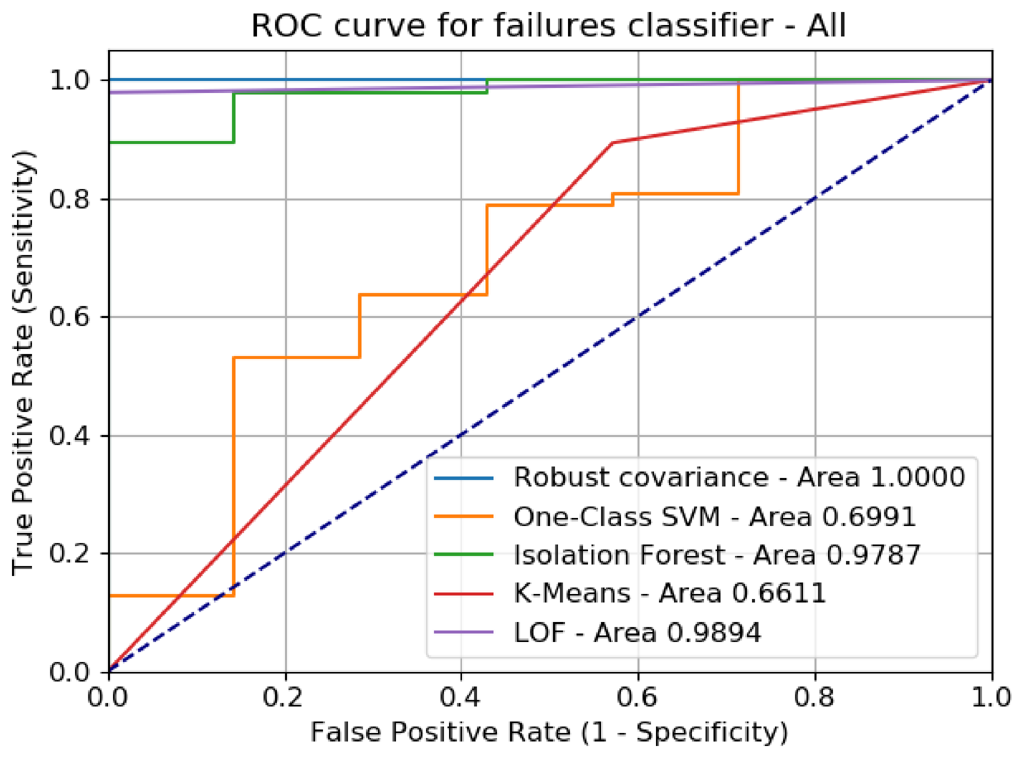


Figure 56. ROC curve and AUC of the algorithm's results for the second mission.

Therefore, the Area Under the Curve (AUC), or the integral of the ROC, gives additional information about the quality of the algorithms analyzed and can be used as a detection performance measure. The AUC can be seen as the probability that an anomaly detection algorithm will assign a randomly chosen normal instance a lower score than a randomly chosen anomalous instance [24]. In particular, from the images above it is clear how the Elliptic Envelope always maintains the same performance in both cases, with an area always equal to one regardless of the chosen threshold, which means that it can correctly classify all the points. In the case of the LOF, instead, we have a slight decrease in performance in the second mission. However, this is not a significant change, the results are still satisfactory, but the best results are still obtained with the Elliptic Envelope.

5.7 Anomalies Localization and Trend Visualization

Once we have found the best results in the Elliptic Envelope as regards the cataloging of the outliers, we use the latter's outputs so as to be able to carry out the graphing of some trends that can be useful to the analysts to carry out a deeper analysis of the areas where too many failures were found.

In Figure 57 is possible to see the areas of central Italy in which an anomalous number of these failures have been found in the first or in second mission. At this point we carry out a comparison of these areas in the two missions, in order to verify the changes.

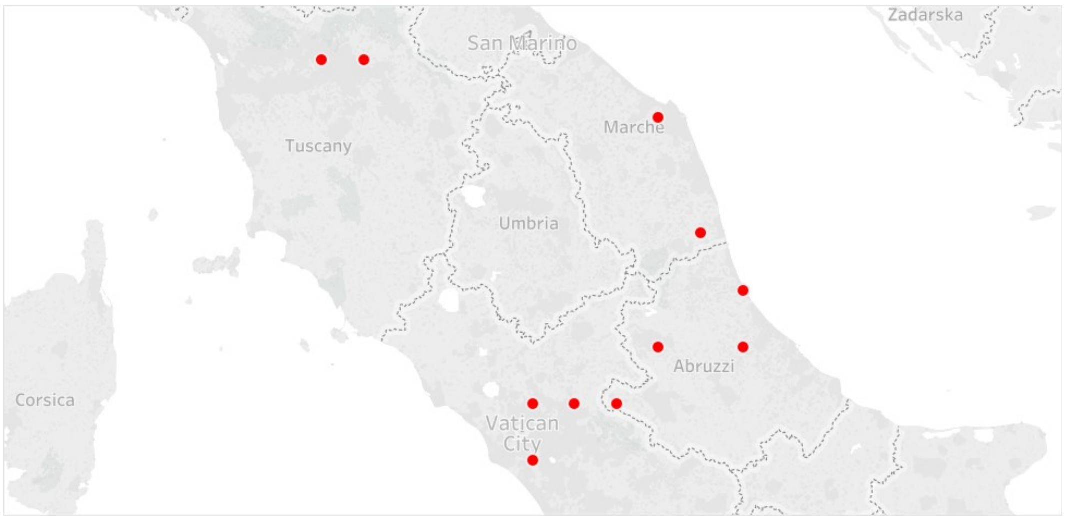
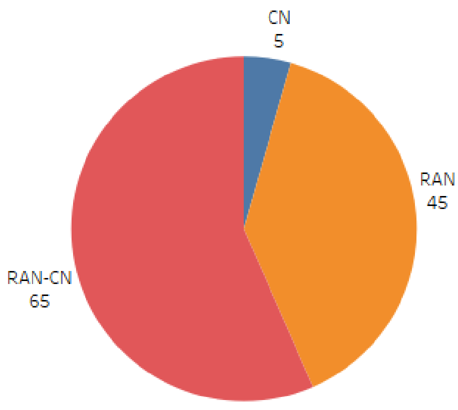
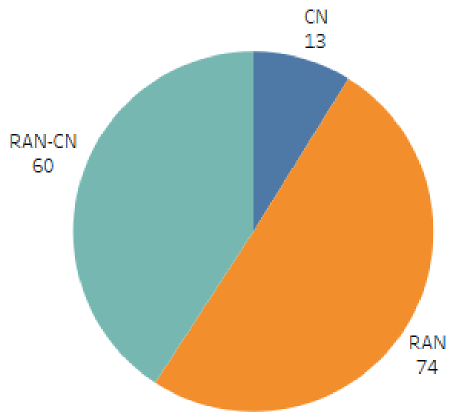


Figure 57. Failures localization

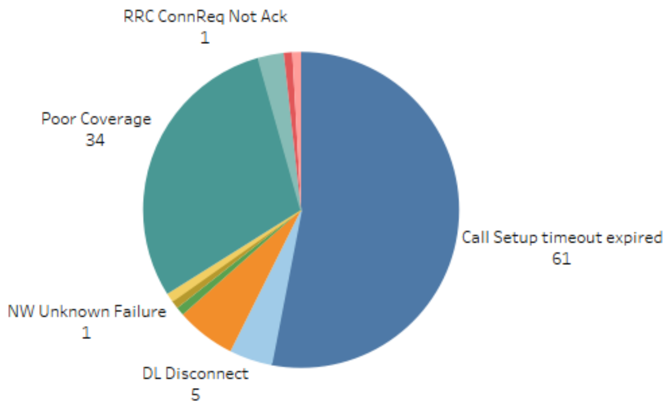
At this point, similarly to what was done in the third chapter, the most significant trends are shown, that are those with reference to the parameters most requested by customers, the mobile operators. Similarly, the following are the analyzes of these trends related to the Root Causes, Failed Side, Category and Phase of the two missions, specifically in the areas where anomalies were found. The comparison of the two missions is therefore carried out not only from the point of view of the number of failures, but also from the point of view of the most significant features.



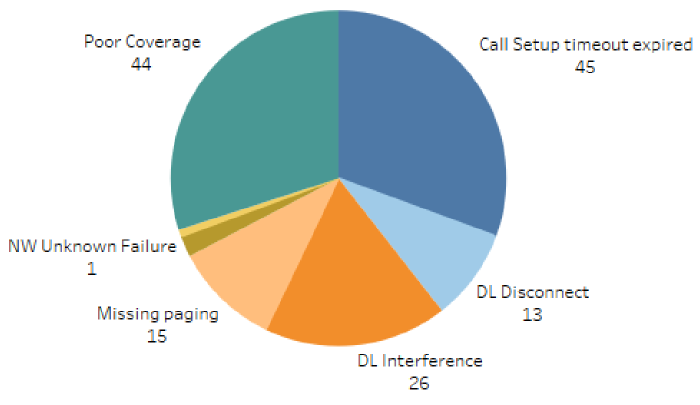


a *b*

Figure 58. Categories' count for first (a) and second (b) mission.



a



b

Figure 59. Root Causes' count for first (a) and second (b) mission.

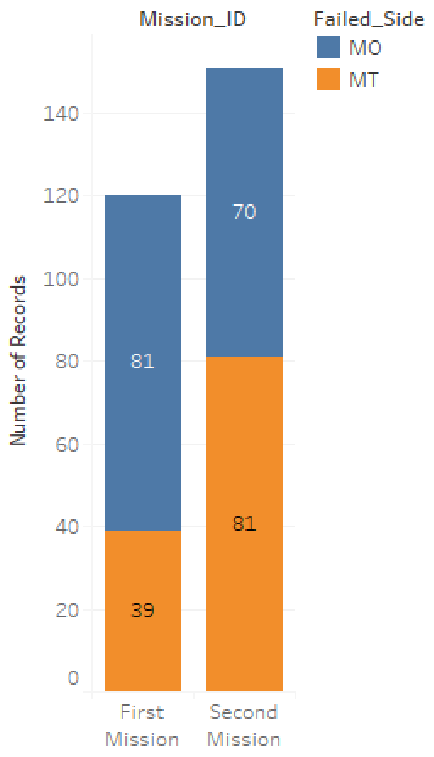


Figure 60. MO & MT comparison between first and second mission

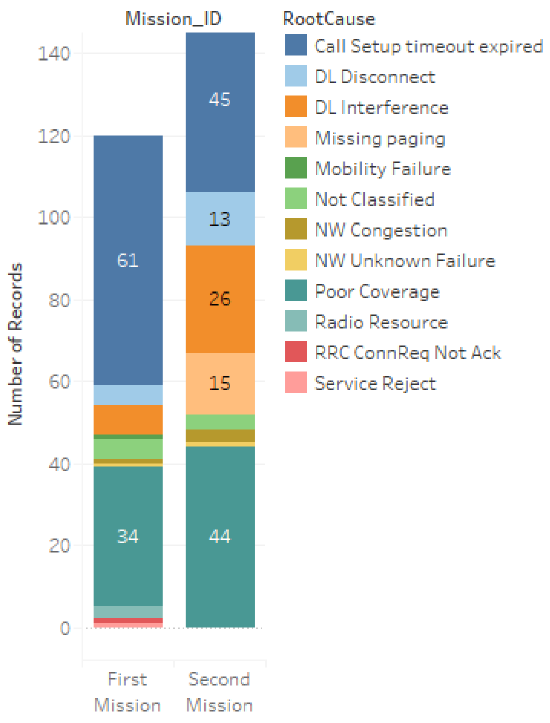


Figure 61. Root Cause comparison between first and second mission.

With the same areas analyzed, 115 failures were recorded for the first mission and 147 for the second one. This is already a fairly interesting fact, which shows that there was no improvement in the infrastructures analyzed in the first mission. However, the fact that there may be many factors that could justify this variation, such as the clogging of the canal during the mission or even the weather, that can sometimes increase the interference, especially in case of rain, but not only of course. Therefore, since it is impossible to consider all the possible factors, we can limit ourselves to making considerations only on this type of data. In Figure 58 we see how the RAN-CN category remains practically unchanged, while there is a considerable increase in the RAN category, thus highlighting a clear problem of access and consequent connection to the network.

In Figure 59, instead, the Root Causes recorded in the two missions are compared. It is easy to see how there is an increase in the number of failures caused by high interference (DL Interference), which could also confirm the hypothesis of high channel clogging, but also of failed paging (Missing Paging), the transition phase from the LTE network to the 2G / 3G legacy network, as also shown in Figure 61.

However, all these failures seem to have been more concentrated on the receiver side (MT) than the first mission, as also shown by Figure 60.

This just conducted is an analysis that aims to verify the state of the zones shown in Figure 57 before and after, comparing the two missions thus, having been carried out these in different periods of the year. Obviously, the analysis can be conducted on a specific area or on a limited set of zones, depending on the needs and considerations that analysts make.

6 CONCLUSIONS

With this thesis work we tried to integrate an emerging technique such as that of anomaly detection in a different context from the ones in which it is usually applied, namely that of telecommunications.

Specifically, the anomaly detection, which we have seen to be a technique forming part of the broad branch of machine learning, aims to identify potential anomalies in a system. The anomalies can be of different nature, they depend strongly on the context and above all there is no common valid pattern for everyone. The analysis often requires some preliminary processing of the data available and the identification of the so-called outliers sometimes does not happen in a clear and correct manner.

In the case we analyzed, however, it seems to have obtained a good level of accuracy.

The dataset at our disposal consists of a collection of logs of failed voice calls in one or more geographical areas that took place at different periods of the year, obtained thanks to benchmarking equipment provided by Reply S.p.a. The initial part of the work consisted in the elaboration of these data, understood as the identification of the main cause of the individual failed calls. This operation was possible thanks to a series of interviews with analysts who were experts in the sector, thus allowing the definition of *ad hoc* logics for the identification of causes. This just described represents the first phase of this thesis work. The selection of the main causes performed in an automatic way was compared with that carried out by the analysts. The results of this comparison are very satisfying and encouraging, being in fact the first time such an approach is tried. For the comparison the features most requested by the customers were considered, and in some cases accuracy levels of 80% were reached, as reported in Table 2. However, there are still some comparisons that fail to reach this percentage, and therefore it is necessary to conduct a greater number of interviews with analysts, in order to carry out a further alignment both on the logic to be used and also on the syntax that is used for the drafting of final reports.

The events thus selected were subsequently grouped into micro geographical zones of about ten square kilometres and then the number of events in the individual micro zones, or pixels, was counted. The detection of anomalies, therefore, was conducted to find areas in which an excessive number of failed calls were recorded with respect to the general pattern of the mission. The goal was to compare the areas thus identified in missions that took place in the same area but in different periods of the year, and see what if something had actually changed, if there was therefore an optimization in the infrastructure or not.

For the purpose, five different anomaly detection algorithms were used, and the approach used was of the unsupervised type, namely a classification of the points made without the use of a classification made *a priori* on the same data, so with unlabelled data. This last choice is justified by the fact that these preliminary classifications are often not available or in any case require some time to be carried out. However, having carried out our analysis on a restricted area, therefore with relatively few points to be analyzed, we carried out this preliminary cataloging (manually done by the analysts) but with the simple purpose of comparing the results obtained by the algorithms with these, thus allowing the development of certain metrics and therefore verify the quality of the cataloging.

Our analysis was conducted on events recorded in central Italy in two different missions, the first one carried out in January 2018 (Figure 36) and the second in July of the same year (Figure 37). The results obtained were reported in Table 7 and 8. In these two tables metrics have been reported relative to the results of the individual algorithms used for the individual missions. These are specific metrics that are created in order to evaluate the quality of the model generated by the single algorithm based on the data supplied to it. These metrics consider the percentage of points correctly classified and not from the model under analysis to the actual classification, in our case the one carried out manually by the analysts.

Analyzing the results shown in the tables it can be seen that all five algorithms used have achieved good results, more in the first mission than in the second, there have been slight decreases in performance in terms of classification.

Among all those reported, however, the one that achieved the best results was undoubtedly the Elliptic Envelope, an algorithm that is often used with relatively small datasets whose data present a Gaussian distribution. Even the Local Outlier

Factor (LOF) was able to obtain satisfactory results, but its configuration was subjected to a series of iterations following which the best one was chosen in terms of accuracy compared to manual classification. Although from the various tests performed the optimal results have been obtained with a number of neighbors between 14 and 17, this is not a setting that should be considered correct in an absolute sense, as the dataset that is analyzed always varies and above all the configuration is obtained following a comparison with the classification made by analysts. The latter, as has already been said, is not always available, rather a study has been conducted on the use of unsupervised algorithms for this reason. The manual classification was made in this case only to develop the mentioned metrics. What we are looking for is an algorithm that is able to obtain satisfactory results almost automatically, and the Elliptic Envelope, following the results obtained, is certainly the best possible model for the type of problem proposed.

Thanks to the results obtained, the possibility of applicability of the anomaly technique to a context such as the highly variable and complex one like that of telecommunications is concrete.

The ultimate goal is to provide analysts with an additional tool not only in the search for the root cause of failed calls, as done in the first phase of the thesis, but also for an optimized and fast search of geographical areas where there is greater concentration of failures. In this way, they will be able to perform in-depth analyzes directly and without wasting resources and time.

However, the data available did not allow us to carry out further types of analysis, since these are only related to bankruptcy events. It would be interesting, therefore, to apply this technique to different data, and therefore to verify the potential of the case. Data that could also include both failed and good calls, or even additional features to those currently supplied, such as modulation, registered interference and so on.

Bibliography

[1] ‘The history of Artificial Intelligence’, *History of Computing, CSEP 590A, University of Washington,* December 2006.

[2] ‘5G Americas’: http://www.5gamericas.org/en/resources/lte-and-lte-advanced-deployments/, Retrieved February 8, 2019.

[3] ‘LTE’: https://it.wikipedia.org/wiki/LTE_(telefonia). Retrieved February 8, 2019.

[4] ‘What is CSFB and SRVCC in LTE?’: http://www.telecomhall.com/what-is-csfb-and-srvcc-in-lte.aspx. Retrieved February 8, 2019.

[5] “Circuit-switched fallback: The first phase of voice evolution for mobile LTE device”, Quallcomm, White Paper, July 18,2013.

[6] “LTE system information blocks – MIB, SIB – 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11”: http://www.rfwireless-world.com/Terminology/LTE-MIB-SIB-system-information-blocks.html. Retrieved February 8, 2019.

[7] ‘SmartBenchmarker: Benchmarking test made easier’: https://www.mobile-network-testing.com/en/products/benchmarking/smartbenchmarker/. Retrieved February 8, 2019.

[8] Sangamesh Ragate, Ndim Hmeidat, Mustfa Aljumaily, ‘ECE-571 Pattern Recognition’, *University of Tennessee, Knoxville, UTK,* December 2015.

[9] Chandola, V., Banerjee, A., and Kumar, V. 2009. “Anomaly detection: A survey”. ACM Comput. Surv. 41, 3, Article 15 (July 2009)

[10] ‘Hierarchical Temporal Memory for Real-time Anomaly Detection’: https://www.slideshare.net/ibobak/hierarchical-temporal-memory-for-realtime-anomaly-detection. Retrieved February 22, 2019.

[11] “Anomalies: A note about findind anomalies”: https://towardsdatascience.com/a-note-about-finding-anomalies-f9cedee38f0b. Retrieved February 22, 2019.

[12] P. N. Tan, M. Steinbach, and V. Kumar, ‘Introduction to Data Mining’. Addison-Wesley, 2005.

[13] A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/A-taxonomy-of-unsupervised-anomaly-detection-algorithms-comprising-of-four-main_fig7_301533547 [accessed 16 Mar, 2019]

[14] Yeung DY, Ding Y. Host-Based Intrusion Detection Using Dynamic and Static Behavioral Models. Pattern Recognition. 2003; 36:229–243. doi: 10.1016/S0031-3203(02)00026-2.

[15] Hawkins S, He H, Williams GJ, Baxter RA. Outlier Detection Using Replicator Neural Networks. In: Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery (DaWaK2000). London, UK: Springer-Verlag; 2000. p. 170–180.

[16] Amer M, Goldstein M, Abdennadher S. Enhancing One-class Support Vector Machines for Unsupervised Anomaly Detection. In: Proceedings of the ACM SIGKDD Workshop on Outlier Detection andDescription (ODD'13). New York, NY, USA: ACM Press; 2013. p. 8–15.

[17] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

[18] 'Fitting an Elliptic Envelope': https://scikit-learn.org/stable/ modules/ outlier_detection.html #fitting-an-elliptic-envelope. Retrieved on March 12, 2019.

[19] Rajesh Kumar, Partha Pratim Kundu, Vir V. Phoha, 'Continous authentication using One-Class Classifiers and their fusion', *IEEE 4th International Conference on Identity, Security, and Behavior Analysis (ISBA)*, 2018.

[20] Markus Goldstein, Seiichi Uchida, 'A Comparative Evaluation of Unsupervised Anomaly detection Algorithms for multivariate data', Dongxiao Zhu, Wayne State University, UNITED STATES, 19 April 2016.

[21] Mennatallah Amer, Markus Goldestein, Slim Abdennadher, 'Enhancing One-Class Support Vector Machines for Unsupervised Anomaly Detection' *ODD'13,* August 11th, 2013, Chicago, IL, USA.

[22] 'K-Means Clustering in Python': https://mubaris.com/posts/kmeans-clustering/. Retrieved on March 12,2019.

[23] 'Accuracy, Precision, Recall or F1?': https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9, Retrieved on March 11, 2019.

[24] Fawcett T. An Introduction to ROC Analysis. Pattern Recognition Letters. 2006; 27(8):861–874. doi: 10.1016/j.patrec.2005.10.010.