الجمهورية الشعبية الديمقراطية الجزائرية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
المدرسة العليا للإعلام الآلي ٠08 ماي 5491٠ بسيدي بلعباس
École Supérieure en Informatique
-08 Mai 1945- Sidi Bel Abbès

MEMOIRE

Pour l'obtention du diplôme **d'ingénieur d'état**
Filière: **Informatique**
Spécialité:: **Système d'Information et Web (SIW)**

Thème

Anomaly Detection using AutoEncoders: The advanced Persistent Threats case

---

Présenté par:
BOUDOUARA Nadjet
LAIB Oumaima

Soutenu le 04 Juillet 2022 Devant le jury composé de:

Dr. KECHAR Mohamed	President
Dr. BENABDERRAHMANE Sid Ahmed	Encadreur
Pr. BENSLIMANE Sidi Mohamed	Co-Encadreur
Dr. KHALDI Miloud	Examinateur

Année Universitaire : 2021/2022

Acknowledgments

*W*e first would like to express our sincere thanks to Allah for helping us and giving us the patience and motivation to complete this work.

Our gratitude and thanks go to Our supervisors **Pr Benslimane Sidi-Mohamede** ,**Dr Benabderahmane Sid Ahmed** for guiding and advising us during the realization of this work, and for the time they devoted to us to realize this work.

We would like to thank our dear parents for giving us the help, encouragement, support and the necessary means and conditions to succeed in our university studies, without forgetting all the members of our family and our close friends.

We close by thanking all my professors and all the staff of the ESI-SBA.

Abstract

Advanced persistent threats (APTs) are long-lasting cyber-attacks aiming at stealing vital data from target businesses. According to security experts, blocking all APTs is impossible, which emphasizes the necessity of early identification and damage limitation studies. Provenance trace mining and whole-system provenance tracking are deemed promising because they can aid in the discovery of causal linkages between activities and the detection of suspicious event sequences as they occur.

We provide here, a machine learning-based solution, that combines between the Auto-Encoders together with the Rule mining, for detecting genuine APT-like cyber attacks using provenance traces that use OS-independent indicators representing process activity.

The APTs and their anomaly scores in the models which were learnt from traces are used to rank anomalous processes. The results are subsequently provided as implications, which would help leveraging the causality for explaining the identified anomalies. Our proposed models were very competitive compared to existing approaches when tested on Transparent Computing program datasets (DARPA).

Keywords: Anomaly detection, Machine learning, Deep learning, web and cyber security, Advanced persistent threats , Big Data, web technologies,Autoencoder.

مُلخص

التهديدات المستمرة المتقدمة (APTs) هي هجمات إلكترونية طويلة الأمد تهدف إلى سرقة البيانات الحيوية من الشركات المستهدفة. وفقًا لخبراء الأمن، فإن حظر جميع APTs أمر مستحيل، مما يؤكد على ضرورة التحديد المبكر ودراسات الحد من الضرر. ويعتبر تعدين المصادر النزرة وتتبع مصادر النظام بأكمله واعدين لأنهما يمكن أن يساعدا في اكتشاف الروابط السببية بين الأنشطة والكشف عن تسلسل الأحداث المشبوهة عند حدوثها.

نحن نقدم تقنية غير خاضعة للإشراف لاكتشاف اعتداءات حقيقية تشبه APT باستخدام آثار المصدر التي تستخدم مؤشرات مستقلة لنظام التشغيل تمثل نشاط العملية.

يتم استخدام الموضعات الحقيقية ودرجاتها في جميع النماذج المستفادة من الآثار لترتيب العمليات الشاذة. وتقدم النتائج فيما بعد بوصفها آثارا تساعد على تعزيز السببية في تفسير الحالات الشاذة المحددة لأنها قابلة للتفسير. تغلبت استراتيجيتنا على الأساليب التنافسية عند اختبارها على مجموعات بيانات برنامج الحوسبة الشفافة (DARPA)

الكلمات المفتاحية : كشف الشذوذ ، التعلم الآلي ، التعلم العميق ، الويب والأمن السيبراني ، التهديدات المستمرة المتقدمة ، البيانات الكبيرة ، تقنيات الويب ، التشفير التلقائي.

Résumé

Les menaces persistantes avancées (APT) sont des cyberattaques de longue durée visant à voler des données vitales aux entreprises cibles. Selon les experts en sécurité, le blocage de tous les APT est impossible, ce qui souligne la nécessité d'études précoces d'identification et de limitation des dommages. L'extraction de traces de provenance et le suivi de provenance de l'ensemble du système sont jugés prometteurs car ils peuvent aider à découvrir des liens de causalité entre les activités et à détecter des séquences d'événements suspects au fur et à mesure qu'ils se produisent.

Nous fournissons une technique non supervisée pour détecter les agressions authentiques de type APT à l'aide de traces de provenance qui utilisent des indicateurs indépendants du système d'exploitation représentant l'activité du processus.

Les vrais positifs et leurs scores dans tous les modèles appris à partir des traces sont utilisés pour classer les processus anormaux. Les résultats sont ensuite fournis sous forme d'implications, ce qui aide à tirer parti de la causalité pour expliquer les anomalies identifiées puisqu'elles sont interprétables. Notre stratégie a battu les approches concurrentes lorsqu'elle a été testée sur des ensembles de données du programme informatique transparent (DARPA).

Mot Clé : Détection d'anomalies, Machine learning, Deep learning, web et cybersécurité, Menaces persistantes avancées, Big Data, technologies web, Autoencodeur.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

- **AI:** Artificial Intelligence
- **ML:** Machine Learning
- **DL:** Deep Learning
- **KNN:** K Nearest Neighbor
- **SVM:** Support Vector Machines
- **NB:** Naive Bayes
- **RF:** Random Forest
- **ANN:** Artificial Neural Networks
- **CV:** Cross Validation
- **AUC:** Area Under the Curve
- **ROC:** Receiver Characteristic Operator
- **APT:**Advanced Persistent Threat
- **NN:**Neural Network
- **ILA:**Inductive Learning Algorithm
- **NLP:**Natural Language Processin
- **MLP:**Multi Layer Perceptron
- **PCA:**Principal Components Analysis
- **DAE:**Denoising Autoencoders
- **CAE:**Contractive Autoencoders
- **MCE:**Mean Square Error
- **MAE:**Mean Absolute Error

- **MSLE:**Mean Square Logarithmic Error
- **GD:**Gradient Descent
- **SGD:** Stochastic Gradient Descent
- **TP:** True Positives
- **TN:** True Negatives
- **FP:** False Positive
- **FN:** False Negative
- **AgTech:** Agricultural technology
- **ICT:** Information and communications technology
- **NDCG:** Normalized Discounted Cumulative Gain
- **IP camera:** Internet Protocol camera,
- **CCTV:** closed-circuit television

Part I

Introduction

CHAPTER 1

INTRODUCTION

1.1 Introduction:

The Internet is ubiquitous, with the major search engines indexing an estimated 1.37 billion unique pages (Ghafir, Prenosil, et al. 2014a), and the world wide web has evolved from a system for serving an interconnected set of static documents to what is now a powerful, versatile, and large platform for application delivery and information dissemination, with businesses increasingly putting critical resources and sensitive data online. Unfortunately, as the web's power and appeal have grown, so has the number and influence of cyber thieves. Criminals are attracted to cybercrime because they have a minimal chance of getting detected and convicted.As a result, a whole business has sprung up to perpetrate cybercrime, and practically every company now faces growing risks to its networks and services. Governments, on the other hand, have discovered that cyberspace can be used to spy on other countries as well as a battleground.

The cost of cybercrime, or illegal activity on cyber infrastructures, is now estimated to be between $100 billion and $1 trillion US dollars each year (Kshetri 2010). Because of the severity of the situation, there has been a lot of interest in exploring techniques that might help minimize the threat. Intrusion detection systems (IDSs) have been presented as a viable technique of detecting and preventing successful computer network exploitation. An intrusion, as described in (Kruegel, Valeur, and Vigna 2004), is a series of connected acts carried out by a malevolent adversary that leads to the breach of a target system. The intruder's activities are deemed to be in violation of a set of security policies.The process of detecting and reacting to harmful activity directed against computing and network resources is known as intrusion detection (ID).

Businesses have tried, with varied degrees of success, to keep malware, spam, and unwelcome invaders at bay for years. Much of the security they've put in place is based on the assumption that most of these assaults would be random and that if an organization's defenses are too difficult to penetrate, the attacker will go for an easier target. According to a technical research by Trend Micro (Ghafir, Prenosil, et al. 2014b), the situation is rapidly changing due to the emergence of targeted assaults (or **advanced persistent threats/APTs**), in which cyber-criminals and hackers target certain enterprises and remain persistent until they achieve their objectives.

Virus scanners, firewalls, and intrusion detection systems were developed to decrease the financial losses caused by cybercrime. As a result, cyber thieves and spies devised increasingly

sophisticated ways to circumvent security systems. An APT is a type of multistep assault that is carried out with more subtlety and is designed to accomplish a certain aim, most commonly espionage. In order to achieve their purpose, APTs employ many phases, much like traditional multistep assaults. APTs, on the other hand, are more likely to rely on so-called "zero-day vulnerabilities" (software weaknesses that aren't publicly disclosed) and advanced methods of attack such as social engineering (Tankard 2011). APTs are the most serious danger to businesses and governments right now (Fossi et al. 2011).

Present detection methods rely on known signatures of attacks, and APTs make extensive use of undiscovered security weaknesses for assaults, these APTs provide a difficulty for current detection systems. A successful APT assault can result in significant economic losses. The estimated financial effect of assaults is the most important factor in deciding whether or not to invest in security measures(Rakes, Deane, and Rees 2012).

Figure 1.1: A an example of an advanced persistent threat attack's life cycle.

1.2 Problematic :

Actual solutions rely on the utilisation of machine learning methods, and more precisely anomaly detection for undertaking the problem of APTs detection. However, one might be faced to several bottlenecks and challenges for figuring out a solution for this particular problem. From those challenges we can point out the most important ones as follows:

- We are dealing here with an imbalanced classification problem since the number of True Positives (a.k.a. APTs) is tremendously lower than the data points with normal behaviour. Consequently, machine learning-wise, the separation between the probabilities of the two classes (normal vs. APTs) is so tiny which makes the miss-classification risk very high.

- Attackers are modifying their attack strategies very frequently, which makes the counter-attack methods sometimes obsolete. Hence the defence models need to be efficient and

adaptive in the same time.

- Normal behaviour is evolving in many fields, and a current definition of normal conduct may not be sufficiently representative in the future.
- The definition of an anomaly varies depending on the application domain. A modest divergence from normal in the medical domain (e.g., changes in body temperature) may be deemed an anomaly, yet a comparable deviation in the stock market domain (e.g., swings in the value of a company) may be considered typical. As a result, transferring a technique established in one area to another is not simple.
- A key challenge would be the frequency of the available labelled data for training/validation of the models utilized by the anomaly detection approaches.
- The data in its original state frequently contains noise that is difficult to filter out since it resembles the genuine abnormalities.

The anomaly detection problems in the case of the imbalanced data, are difficult because of the aforementioned challenges. In practice, most of the anomaly detection systems that have been developed undertake a specific case of application. Various factors, such as the nature of the data, the availability of labelled data, the sort of outliers to be identified, ... , influence the formulation of the solution. For instance, the application domain in which the outliers have to be discovered, could determine these characteristics. Researchers have borrowed ideas from a variety of fields, including statistics, machine learning, data mining, and information science.

1.3 Objectives:

Our objective here is to come up with a solution that considers the problematic and the challenges that we have previously pointed out. To be more specific, the proposed solution need to respect the following milestones:

- Implementing a big-data model for storing the attack logs.
- Implementing a deep learning model using auto encoders to discover the outlier patterns.
- Implementing a meta-learning model that combines several types of classification methods.
- Designing a generative adversarial neural network to produce synthetic cyber-attacks.

1.4 Overview of our solutions :

0. **Artificial Intelligence / Machine Learning Algorithms:**
    - We have developed several machine learning anomaly detection methods for identifying APTS:
      -Auto-encoders.
      -Rule mining.
      -OC-SVM.

-Elliptic Envelope.
-LOF (Local Outlier Factors).
-Isolation Forests.
These approaches are based on both supervised and non supervised AI paradigms, where a decision model is created using an input database.

0. **Web technologies**

- We have developed a web interface that interacts with a core module representing the different aforementioned machine learning methods. This web module, has the following activies: Loading the data, running the classification algorithms, and displaying the results.
- Coding abilities are required for the creation of a website. A website is divided into two parts: front end and back end. Front end is where the server works and requests are seen, such as the interface, buttons, pictures, and so on. Back end is the opposite, where everything is hider sent to save or request data, for example.
- What are the technologies?
  - **Front End :**
    * HTML5 (HyperText Markup Language Version 5)
    * CSS3 (Cascading Style Sheets Version 3)
    * Javascript
    * jQuery
    * Bootstrap
  - **Back End :**
    * Django Framework v3.2.5
    * Python v3.8.5

1.5 Challenges :

- Big Data : We need to deal with large volume of data and implement a model that is scalable.
- Streaming : APTs are generated in real time, by consequence the model needs to deal with streaming analysis.
- Imbalanced problem : The true positives in the database are rare, hence the implemented model should have a certain degree of effectiveness regarding the imbalances data issue.

1.6 Project follow up :

This project is a scientific collaboration between New York university (Sid Ahmed Benabderrahmane) and the National School of Computing of Sidi Bel Abbes (ESI-SBA) (Sidi Mohammed BENSLIMANE).

1.6.1 External collaboration:

The supervision outside of the school with the project manager Sid Ahmed Benabderrahmane was very active, with several email exchanges and meetings per week. The supervisor checked consistently the progress of the performed tasks. He also participated in the decision-making on the conceptual aspects.

1.6.2 Supervision at school:

Our main objective during this internship was to carry out a complete work and in conformity with the qualities of a work carried out by a state engineer of the ESI-Sba, therefore the supervision on the side of the ESI-Sba was essential. Follow-up by the pedagogical supervisor (Mr. Sidi Mohammed BENSLIMANE) was done through reports summarizing all the tasks carried out and to be followed, the results obtained and the theoretical justifications concerning them. The role of the ESI-Sba supervisor was to validate the main guidelines of the internship, validate the theoretical concepts, proofread and validate the dissertation and above all to ensure that the work carried out complied with the requirements imposed by the ESI-Sba.

Part II

Background

CHAPTER 2

ARTIFICIAL INTELLIGENCE (AI)

In this chapter, we will give in detail how algorithms of Artificial Intelligence would work and the steps of the website creation. Since there are multiple models & algorithms for the website, we will focus on the back end since it contains the whole process and the main modules.

2.1 Artificial Intelligence:

Artificial intelligence is the simulation of human mental processes by computers that are programmed to think like people and act like them (AI).

Any machine that demonstrates human-like traits such as learning and problem-solving is referred to as a humanoid machine.

Artificial intelligence's ideal characteristic is its ability to assess and execute activities that have the best chance of achieving a given goal. Machine learning is a subset of artificial intelligence that relates to the concept of computer systems learning from and adapting to changes without the need for human intervention. Deep learning methods aid automatic learning by absorbing large amounts of unstructured data such as text, images, or video.

The majority of people automatically think of robots when they hear the words artificial intelligence. This is because major motion pictures and books show humanoid robots wreaking havoc on the world. Nothing could be farther from the truth, though.

Artificial intelligence is founded on the idea that human intellect may be characterized in such a manner that a computer can simply imitate it and carry out tasks ranging from simple to complicated. Simulating cognitive human processes includes artificial intelligence goals. To the degree that they can be concretely described, researchers and developers in the field are making remarkably quick progress in simulating processes such as learning, reasoning, and perception. Some think that in the near future, inventors will be able to create systems that can learn and reason about any subject faster than humans can. Others, on the other hand, remain skeptical, claiming that all cognitive activity is loaded with value judgments based on the human experience.

Old artificial intelligence standards become obsolete as technology advances. Because these processes are now considered basic computer tasks, machines that solve fundamental arithmetic or recognize text using optical character recognition are no longer referred to be artificial intelligence.

Artificial intelligence (AI) is continually evolving to assist a wide range of industries. Machines are wired using a multidisciplinary process based on mathematics, computer science, linguistics, psychology, and other fields.

2.1.1 Artificial Intelligence Applications :

The applications of artificial intelligence are limitless. The method may be used to a variety of industries and areas. In the healthcare industry, AI is used for medicine delivery and different therapies in patients, as well as operating room procedures (Bonner 2019).

- **AI in healthcare :** Modern healthcare is being revolutionized and strengthened by artificial intelligence-based technology that can comprehend, learn, and act, whether they are used to find novel correlations between genetic codes or to guide surgical robots. The study specifically focuses on the three newest applications of AI in healthcare: AI-driven drug development, clinical trials, and patient care, and the results also suggest that AI-assisted clinical trials are able to handle enormous amounts of data and generate extremely precise outcomes (Shaheen 2021).

- **AI in games :** Artificial intelligence (AI) in gaming focuses on using AI to develop more responsive, adaptable, and difficult games. ai used in games as NPCS (These are in-game characters that behave intelligently, as though they were being controlled by actual players. Artificial intelligence engines and algorithms control how these characters behave. Decision trees are frequently employed to direct the behavior of these NPCs.), Pathfinding(Getting from one place to another is a part of pathfinding. The most crucial aspect of pathfinding is the overall gaming environment. As you explore the game area, the AI can create the game's terrain or its world. The AI may construct the terrain based on feedback it receives from your actions, playing style, in-game choices, appearance, and tactics.) ,Decision-making(AI will allow your choices to have a greater influence on the gameplay. For instance, in Red Dead Redemption 2, NPC interactions and conduct are influenced by things like the sort of hat you are wearing or if you have blood stains on your clothes. Your choices might affect the whole game world because there is such a vast matrix of potential outcomes.) ,Data mining(enables game developers and studios to do data mining on player behavior to better understand how players interact with the game.),ect .... [1].

  There is enormously games are using AI below we have some examples [2]:

  - The Last of Us : The Last of Us' AI is based on the concept of Skills and Behaviours: skills are high-level notions of what a character may be doing. A Hunter could investigate a sound, hide behind cover, or flank the player. Meanwhile, the Infected may be meandering around the map or pursuing an opponent [3].

[1] https://www.engati.com/blog/ai-in-gaming
[2] https://minditsystems.com/ai-games-video-games-with-artificial-intelligence/
[3] https://www.gamedeveloper.com/design/endure-and-survive-the-ai-of-the-last-of-us

Figure 2.1: The Last of Us video that uses AI.

– F.E.A.R [4]: is another horror video game that used AI to interact with the players depending on their behavior.

Figure 2.2: F.E.A.R video game that uses AI.

- **AI in Transport: :** Many studies have proved the benefits of artificial intelligence in transportation. One example is converting roadside traffic sensors into a smart agent that identifies accidents automatically and forecasts future traffic conditions. It is divided into two types: supervised and unsupervised learning methods. Support Vector Machine (SVM), Probabilistic Neural Network (PNN), Radial Basis Network (RBN), and Decision Tree are examples of supervised approaches (Abduljabbar et al. 2019).
  Also Many big automakers are developing their own autonomous vehicles and driving features, but we'll focus on relatively new tech businesses and startups that have sprung from the concept of self-driving vehicles. The following firms are at the forefront of autonomous car technology, whether for public transit, ride sharing, or personal needs. As example Motional , Refraction AI ,Optimus Ride , Waymo and others [5] .

- **AI and NLP :** Chatbots are a type of bot that has previously appeared on chat networks. The user can interact with them using graphical interfaces or widgets, and this is the trend. Chat bots are intelligent systems that use artificial intelligence (AI) and natural language processing (NLP) techniques to communicate. Chat bots are intelligent systems that use artificial intelligence (AI) and natural language processing (NLP) techniques to communicate. It features an intuitive user interface and responds

[4]https://minditsystems.com/ai-games-video-games-with-artificial-intelligence/
[5]https://builtin.com/artificial-intelligence/artificial-intelligence-automotive-industry

to questions about the examination cell, admission, academics, user attendance and grade point average, placement cell, and other miscellaneous activities (Lalwani et al. 2018). As an example, Roof Ai is a chatbot that assists real estate marketers in automating lead generation and lead assignment via social media. The bot finds possible leads on Facebook and then answers nearly instantly in a nice, helpful, and conversational tone that is similar to that of a human person. Roof Ai enables potential leads to offer more information based on user input before automatically allocating the lead to a sales representative.

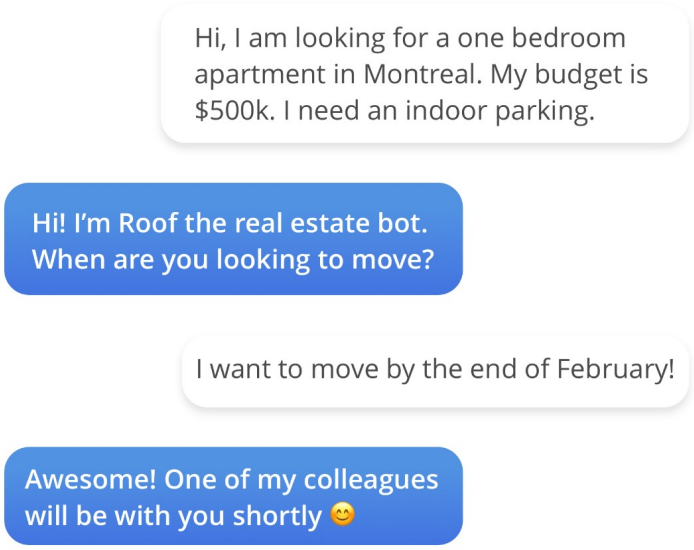


Figure 2.3: Roof Ai: An AI-based intelligent chat bot used in e-commerce.

- **AI in Computer Vision :** Emerging technologies such as computer vision and artificial intelligence (AI) are expected to take use of huge data availability for active training, resulting in operational real-time smart devices and predictable models. Computer vision and AI driven food business refers to the phenomena of using vision and learning approaches to improve the food industry. It describes the growing interest in the AgTech business, which uses computer vision and AI to produce sustainable food to feed the future (Jin, Tan, and Jiang 2020).

- **AI in Smart cities :** As society advanced in the digital era, so did the need for human establishments to keep up. Keeping up with technology breakthroughs has become the only way to go. With this goal in mind, solutions to urban problems must be'smart.' Using modern ICT allows us to create smart solutions. The most current innovation in Artificial Intelligence has transformed intelligent solutions. This innovation in artificial intelligence has improved the way cities battle critical risks to safety and security (Géron 2017). The worldwide urban population is anticipated to reach 66 percent or 70 percent by 2050. This rate of urbanization will have a significant influence on the environment, management, and security of cities. Many governments have advocated the notion of smart cities to effectively manage resources and optimize energy usage in order to deal with the dramatic rise in urbanization. Smart cities initiatives may specifically address the issue of ensuring the green environment by creating and implementing low carbon emission technology. Many governments (e.g., the United States, the European Union, Japan, and others) throughout the world have planned and implemented smart cities

Figure 2.4: Video analytics with deep learning for vehicle detection (Boesch 2022)

Figure 2.5: Object Detection in Smart Cities to recognize dangerous situations (Boesch 2022) .

programs to efficiently address potential emerging difficulties. To satisfy the criteria of a smart city, effective use of information and communication technologies (ICTs) is required to effectively manage data (Ullah et al. 2020). Smart City Computer Vision solutions frequently integrate pre-installed network cameras (IP cameras or CCTV cameras) to offer input for real-time video analytics using AI models.

2.1.2 Machine Learning:

2.1.2.1 What is Machine Learning?

- **Definition 1 :**According to Tom M. Mitchell, "machine learning is the study of computer algorithms that enable computer programs to be automated via experience." Artificial intelligence includes a component called machine learning. We expect AI to be performed in a variety of ways, including machine learning. Machine learning uses datasets to analyse and compare data in order to uncover similar patterns and explore subtleties(Mitchell 1997).

- **Definition 2 :**Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed (Samuel 1959).

2.1.2.2 Types of Machine Learning Systems :

Because there are so many distinct types of Machine Learning systems, it's helpful to group them together into broad groups based on (Géron 2017) :

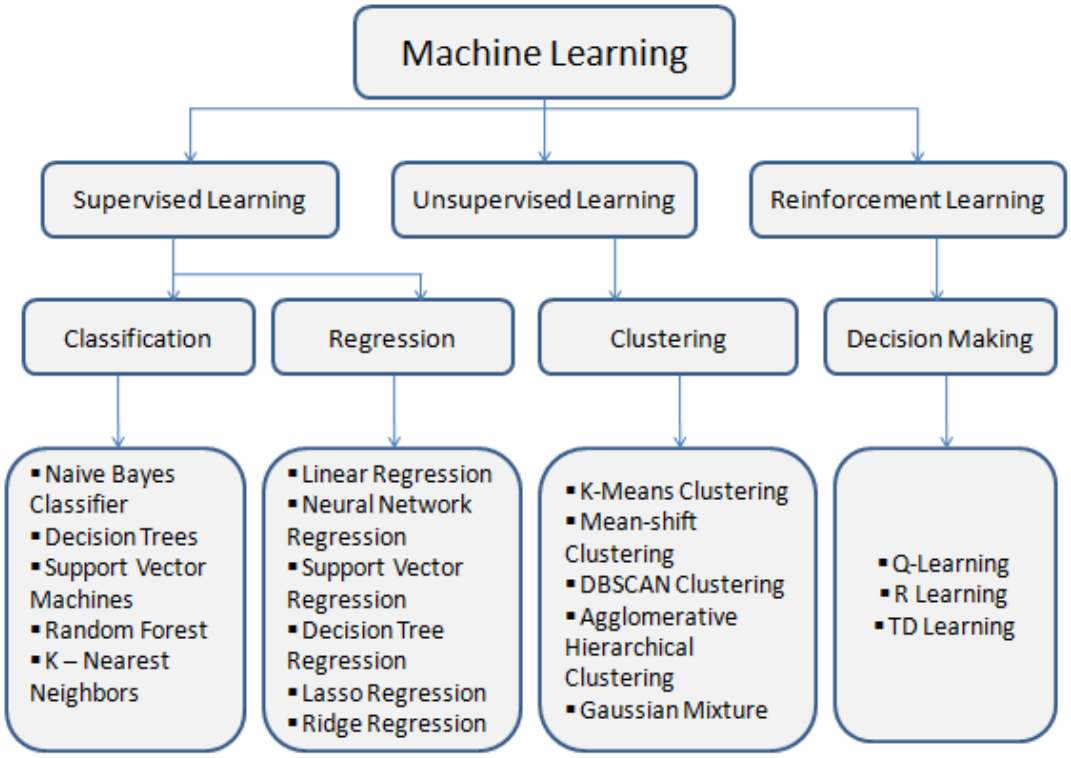


Figure 2.6: Types of Machine Learning methods.

- Whether they are trained with or without human supervision (supervised, unsupervised, semi-supervised, and Reinforcement Learning).
- Whether or whether they are able to learn in little chunks on the fly (online versus batch learning).
- Whether they function by simply comparing incoming data points to previous data points, or by detecting patterns in the training data and developing a prediction model, they work in the same way that scientists do (instance-based versus model-based learning)

2.1.2.3 Machine learning applications :

If, for example, you provide a machine learning model with a large number of songs you enjoy and their audio information (dance-ability, instrumentality, tempo, or genre). It is tough to automate (depending on the supervised machine learning model employed) and build a suggestion to advise that you play music in the future, as well as Netflix, Spotify, and other firms, with a high chance rate.
Another machine learning example that uses web data is recommendation using users traces. Indeed, web platforms such Amazon or Youtube use the web traces and history of navigation for recommending future content (items to purchase in the case of Amazon, or Videos to view in the case of Youtube) using machine learning algorithms.

2.1.3 Deep Learning :

2.1.3.1 What is deep learning :

Deep learning (also known as deep structured learning or differential programming) is a subset of machine learning that focuses on data representation and adds successive learning layers to improve the input data's meaningful representation (Benabderrahmane, Mellouli, and Lamolle 2018).

2.1.3.2 Deep learning applications :

Deep Learning has aided tremendous development in a variety of fields. There are several uses, and the following list is by no means complete:

- Objet recognition in images
- Speech recognition
- Automated translation
- processing of signals
- Web mining
- Bioinformatics

2.1.3.3 Deep learning models :

Several Deep Learning models have been presented, with the following models receiving special attention:

- **Autoencoders (AEs)**
- Convolutional neural networks (CNNs).
- Recurrent neural networks (RNNs)
- Gated recurrent unit (GRU)

Because Auto Encoder is the main algorithm that we developed in or project, this is why we would like to give more details compared to the rest of the algorithms.

2.2 What's an autoencoder?

- **Definition 01 :**Neural networks come in different forms and sizes, and their input and output data types are frequently used to define them. Convolutional Neural Networks, for example, are used to create image classifiers. They take photos as input and provide a probability distribution of the classes as an output.

  Autoencoders (AE) are a type of neural network in which the input and output are the same. They function by compressing the input into a latent-space representation and reconstructing the output from this representation.

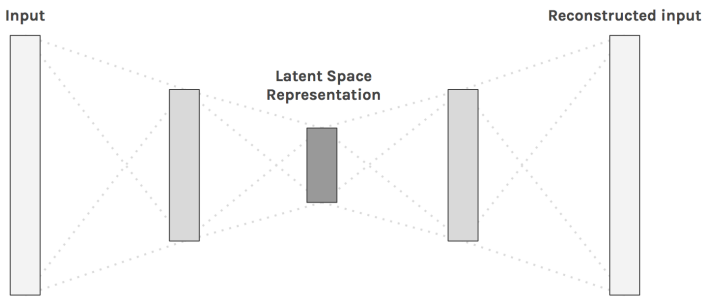


Figure 2.7: Simple Autoencoder architecture The input is compressed and then reconstructed

- **Definition 02:**Autoencoders are a form of neural network design that produces the same output as the input. Unsupervised training is used to teach autoencoders how to learn extremely low level representations of input data. The real data is then projected using these low-level attributes. A regression job in which the network is asked to predict its input is known as an autoencoder (in other words, model the identity function). These networks have a few neurons in the centre that act as a bottleneck, driving them to build efficient representations that compress the input into a low-dimensional code that the decoder can use to replicate the original input.

2.3 Auto encoder's architecture :

A typical autoencoder architecture comprises of three main components:

- **Encoding Architecture :** The encoder architecture is made up of a sequence of layers with fewer nodes that eventually reduce to a latent view representation.
- **Latent View Representation :** The lowest level space in which inputs are decreased but information is kept is known as latent view.
- **Decoding Architecture :** The decoding design is a mirror copy of the encoding architecture, except that the number of nodes in each layer grows, and the result is virtually identical.

2.4 Why do we need AutoEncoders :

Many people wonder why we should learn and utilize Autoencoder if we already have PCA. Is it just due to its ability to work with non-linear data? So the answer is no, because Autoencoder has a variety of applications ranging from computer vision to time series forecasting, in addition to coping with non-linear data.

- **Non-linear Transformations:**it can learn non-linear activation functions and multiple layers.
- **Convolutional layer:** it doesn't have to learn dense layers to use CNN or LSTM.
- **Higher Efficiency :**More efficient in model parameters to learn several layers with an autoencoder rather than learn one huge transformation with PCA.

- **Multiple transformations :**An autoencoder also gives a representation as to the output of each layer, and having multiple representations of different dimensions is always useful. An autoencoder lets you use pre-trained layers from another model to apply transfer learning to prime the encoder and decoder.

2.5 Components of AutoEncoders :

The encoder and decoder are the two halves of the autoencoder. Three components are also mentioned in some publications, with the third being code, which serves as a middleware between the two.

- **Encoders:**The input is compressed into a latent space representation. The encoder layer compresses the input picture into a smaller representation; the compressed image now resembles the original image but is not the original image. own as a program
- **Code :**An encoder, also known as a bottleneck layer, maps input space into lower-dimensional latent space (represented as z in architecture). It is currently a lower-dimensional, unsupervised representation of data. The compressed input given to the decoder is represented by code.
- **Decoders :**The decoder restores the original picture of the same dimension from the encoded image. The decoder transfers data from the lower latent space to the reconstruction phase, when the output X bar's dimensionality equals output X. However, there is lossless compression in the case of Image compression, but lossy compression in the case of Autoencoders, thus what occurs is that it compresses and uncompresses the input. It attempts to get close to the input when it uncompresses, but the result isn't the same.

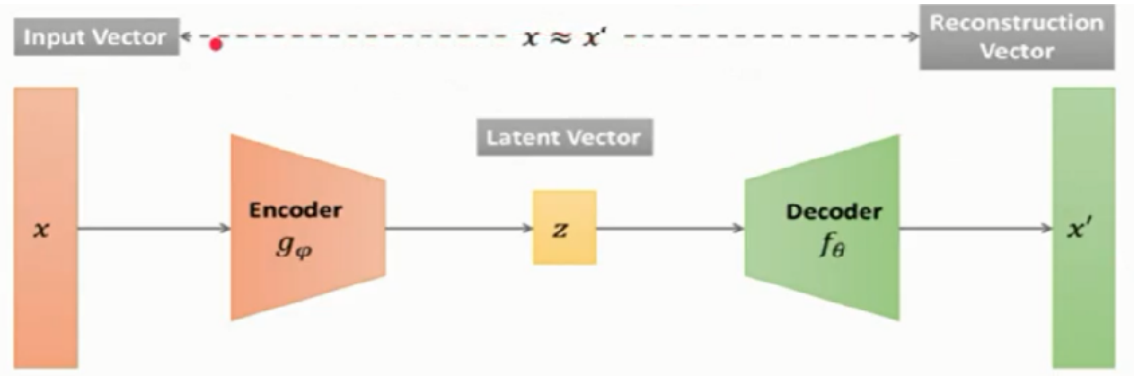


Figure 2.8: Architecture of an AutoEncoder.

2.6 Types of autoencoders:

Here are five popular autoencoders that we will discuss:

- **Undercomplete Autoencoders:** Undercomplete autoencoder takes in an image and tries to predict the same image as output, thus reconstructing the image from the compressed bottlen

- **Sparse Autoencoders:** Sparse autoencoders are similar to the undercomplete autoencoders in that they use the same image as input and ground truth. However—

  The means via which encoding of information is regulated is significantly different.

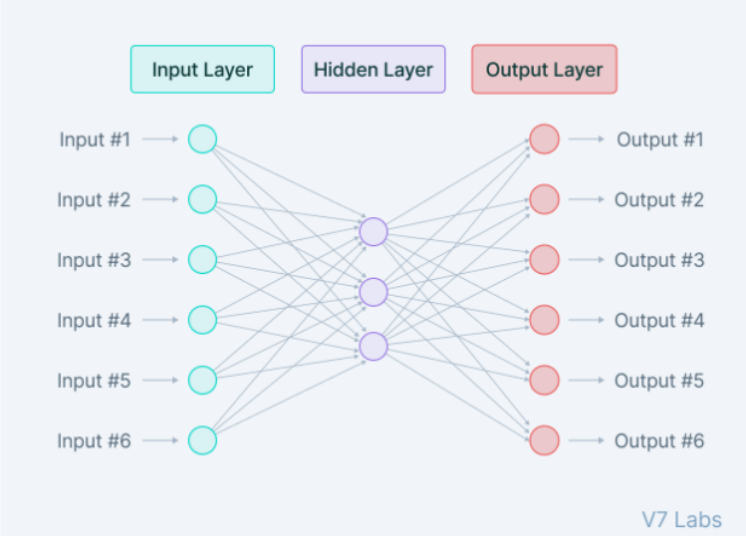


Figure 2.9: Sparse autoencoders

- **Contractive Autoencoders**Similar to other autoencoders, contractive autoencoders perform task of learning a representation of the image while passing it through a bottleneck and reconstructing it in the decoder.

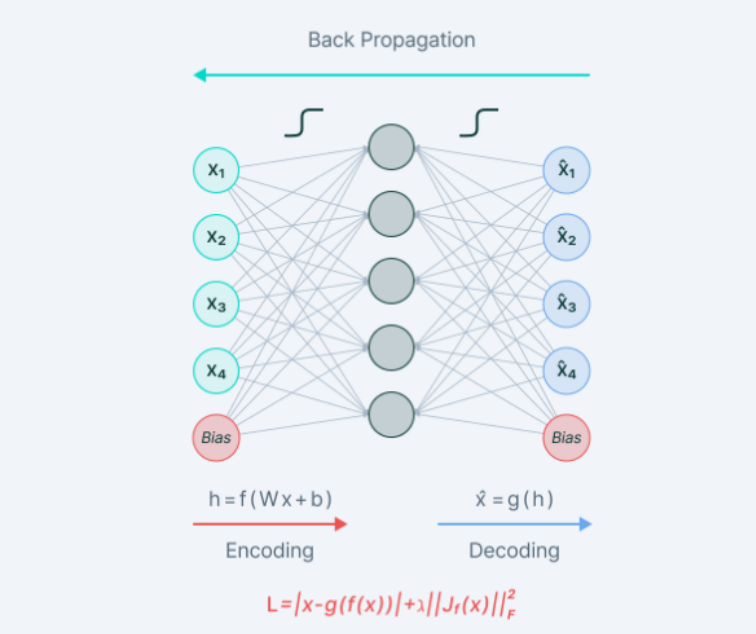


Figure 2.10: Contractive Autoencoders

- **Denoising Autoencoders:** Denoising autoencoders, as the name suggests, are autoencoders that remove noise from an image.

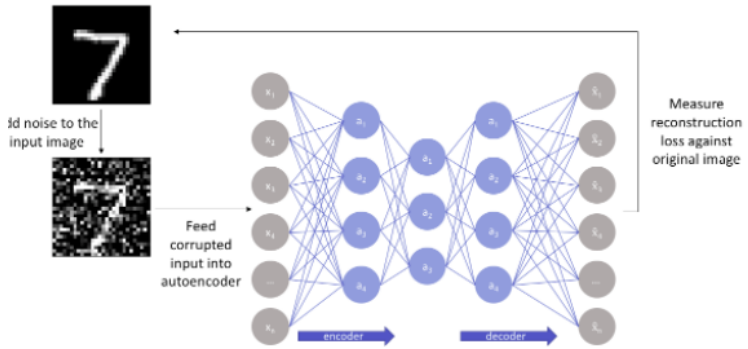


Figure 2.11: Denoising Autoencoders

- **Variational Autoencoders:** Standard and variational autoencoders learn to represent the input just in a compressed form called the latent space or the bottleneck. This is what a variational autoencoder would learn from the input:

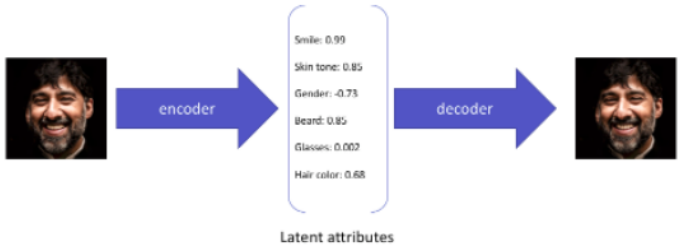


Figure 2.12: Variational Autoencoders

2.7 Properties of AutoEncoders:

Let us look at the important properties passed by autoencoders:

- **Unsupervised :** They do not need labels to train on.
- **Data specific:** They can only compress data in the same way that they were trained on. An autoencoder trained on the human face, for example, may struggle with photos of contemporary skyscrapers. This clarifies the distinction between an auto-encoder and an mp3 compression technique, which is based only on sound assumptions.
- **Lossy:** Because autoencoders are lossy, the decompressed output will suffer.

CHAPTER 3

WEB TECHNOLOGIES :

3.1 Technical notions :

In this chapter, we discuss most of the technical terms and definitions.

3.1.1 Website:

A British physicist devised the World Wide Web in 1989, while Tim Berners-Lee was working on CERN (WWW). The Web was created to address the requests of scientists in colleges and institutions throughout the world for automated information exchange.

A personal website, a business website for a corporation, a government website, a website for an organization, and so on are all examples of websites. Websites can be created by an individual, a company, or another organization, and they usually focus on a single topic or purpose. Every website can have a clickable link to any other website, allowing consumers to blur the lines between different websites.

Some websites demand users to register or subscribe in order to view the material. Business websites, news sites, academic newspapers, gaming websites, file-sharing websites, discussion boards, web-based emails, social networking sites, websites that give real-time inventory data, and websites that offer various services are examples of such web pages.

3.1.2 Responsivity:

Responsive Web Design (RWD) is a web development method that dynamically changes the appearance of a website based on the size and orientation of the display used to view it. RWD solves the difficulty of developing a variety of devices for users, ranging from mobile phones to large desktop monitoring systems.

3.1.3 API (Application Programming Interface):

The API stands for application programming interface, which is a piece of software that allows two apps to communicate with each other. When you use a Facebook app, you'll always utilize an API to send an instant message or check the weather on your phone.

When you use a mobile phone application, it connects to the Internet and sends data to a server. The server then collects, plays, and sends this data back to your phone. The program then interprets and reads this information. This is an API, and it's used for everything.

3.1.4 MVC (Model View Controller) & MVT (Model View Template):

Model view control (MVC) is an architecture pattern that divides a single program into three basic logical parts: the model, the view, and the controlling system. Each component is tailored to a certain area of application development. MVC is the most popular web development framework for scalable and extendable industry standard applications.

- **Model :** All of the user's data logic is compatible with the model's component. This can be data sent between view components and controller components, or any other business logic data. Customer information is retrieved, managed, and updated from and to the database by the object of the customer in a database, for example.
- **View :** All of the application's UI logic is handled by the view component. User interfaces containing user components like as text inputs, drop-downs, and so on are included in the customer view.
- **Controller:** The controls function as a link between the Models and View components, allowing them to conduct business logic and accept requests, alter data using the Model component, and interact to accomplish the desired outcome. The customer controller, for example, is in charge of all interactions and inputs in the customer view, as well as updating the customer model database. Client data is viewed using the same controller.

Model View Template, often known as MVT, is a similar design pattern to MVC. MVT is a design pattern that separates code into three components in the same way as MVC does.

- **Template :**The distinguishing part between MVT and MVC. Templates function as the display layer and are mainly the HTML code returning the data. The file contents might be static dynamically

3.1.5 ORM (Object-relational mapping):

In computer science, object-relation-mapping is a programming method for transforming data across incompatible types of systems using target-oriented programs. A "virtual object database" is created, which may be utilized as a programming language. Both free and commercial solutions map objects, and some programmers choose to build their own ORM tools.

Data management tasks are performed on objects that are practically never scaled in object-driven programming. With zero or more numbers and zero or more addresses, an individual book of addresses, for example, should be considered. This may be modeled in an object-oriented implementation of a personal object with an attribute/field to store any element in the data that is contained in the entry: a person's name, phone number, and address list. "PhoneNumber Objects" and similar items would be included in the phone number list. Each book entry will be treated as a single item in the programming language. Additional methods, such as returning the phone number, address, or other information you requested, may be connected to the object.

By contrast, many traditional databases, such as DBMS, are not object-oriented and can only store and analyze scalar data like numeric tables and textual. The programmer either transforms (and then recovers) or merely utilizes the scalar value in the program to break down an object value into smaller groupings of storage data. The first method is to carry out object mapping.

The key difficulty is to list the objects in an atomized form that can be kept in a database while keeping the objects' attributes and connections so that they may be reloaded as objects if necessary. The items will be persistent if this approach of storing and recovery is used.

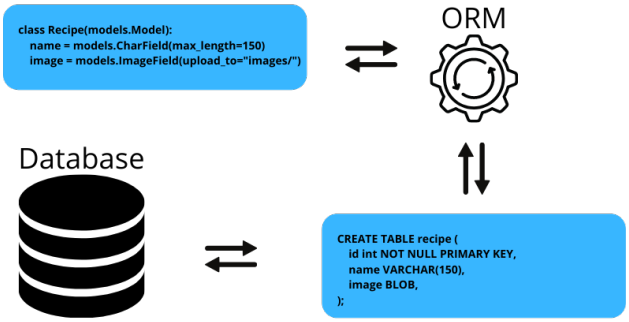


Figure 3.1: Django Model-based database using ORM

3.1.6 Front End :

The front end employs technologies such as the language of hypertext markup, JavaScript, and case sheets A combination of the front end and the back end (CSS). Developers are pioneering the creation and development of user-friendly websites and apps, including buttons, menus, pages, links, graphs, and more.

- **HTML:** Hypertext Markup Language is used to provide the site's appearance and functionality. HTML5.2, the most recent version, was released in late 2017. To facilitate interoperability, the new edition adds more tools for web app development.
- **CSS:** Developers may create stunning and dynamic website designs with the freedom and accuracy of cascading sheet style.
- **JavaScript:** On static web pages, dynamic components can be built. This event's lingua franca.
  It allows developers to access various parts and respond to server events on the main HTML page.
  Angular, Ember, Backbone, and React are some of the most popular front-end frameworks.
  These framework models assist developers in meeting the growing need for enterprise software without sacrificing quality.

3.1.7 Back End :

The server-side background consists of a server that supplies data on demand, a channeling application, and an information database.
The server-side background consists of a server that supplies data on demand, a channeling application, and an information database.
The information will be stored in a database on the server when you choose the item, add it to your shopping cart, and complete the transaction.
When the customer checks the status of the package a few days later, the server pulls up the necessary information, updates it, and displays the tracking data.

3.1.7.1 Back End Tools :

Back-end developers' key priorities include creating applications that locate and deliver data on the front end.

The apps can use a variety of frameworks or languages, including Ruby on Rails, Java, C++/C#, Python, and PHP.
Back end-to-service (BaaS) providers have been a viable alternative for many years. They're especially beneficial when working on mobile apps on a tight deadline.

3.1.8 Database :

Oracle, Microsoft SQL Server, Teradata, IBM DB2, and EnterpriseDB are among the most popular databases used by enterprises.
Other popular databases such as MySQL, NoSQL,mongoBD, and PostgreSQL are also available.

- **MongoDb :** MongoDB is a document-oriented NoSQL database that debuted in the middle of the 2000s. Huge amounts of information are stored there.

  MongoDB doesn't rely on tables and columns like a conventional SQL relational database does. Collections and documents are used to store data.

  Documents are value/key pairs that serve as the fundamental data unit. Collections, on the other hand, are made up of groups of documents and functions. They are analogous to tables in traditional relational databases [1].

3.1.8.1 Formats :

- **JavaScript Object Notation (JSON) :** is a text-based, schema-less representation data structure that uses ordered lists and key-value pairs. String, integer, Boolean, list, object, and null are the only value types available in the general data format JSON. JSON is an excellent choice to communicate data across language barriers because, despite the notation being a subset of JavaScript, these kinds are present in all popular programming languages (Freeman 2019). JSON data is saved in files with the.json extension.

  Here's an example of JSON-encoded data:


```
[
  {
    "": 0,
    "uuid": "aae684ad-73de-3159-aa8e-68256e6af943",
    "label": "AdmSubject::Node"
  },
  {
    "": 1,
    "uuid": "9ae560c3-57ca-316e-b3c4-a485c3105ccc",
    "label": "AdmSubject::Node"
  },
  {
    "": 2,
    "uuid": "97e2d3c9-0379-382a-86c6-c3c2e17dc6eb",
    "label": "AdmEvent::Node"
  },
```


[1] https://datascientest.com/mongodb

- **Comma Separated Values (CSV) :** A CSV file is a plain text file that contains a list of data separated by commas. These files are frequently used to exchange data between apps. Databases and contact managers, for example, frequently support CSV files. The structure of a CSV file is rather basic. A comma separates each item in a list of data(HOFFMAN 2018).

  Here's an example of CSV file :

```
,uuid,label
0,aae684ad-73de-3159-aa8e-68256e6af943,AdmSubject::Node
1,9ae560c3-57ca-316e-b3c4-a485c3105ccc,AdmSubject::Node
2,97e2d3c9-0379-382a-86c6-c3c2e17dc6eb,AdmEvent::Node
3,2745271d-40a8-3b57-84bb-1e1c023164c0,AdmNetFlowObject::Node
4,9517d8bb-2f29-3d25-b816-27d19e765093,AdmNetFlowObject::Node
5,662a2016-e0d3-336e-9bbd-70ea17d5afe4,AdmNetFlowObject::Node
6,6a7f6f8e-8b13-330b-8932-315172bd3cf5,AdmEvent::Node
7,96377f18-5cc4-31b4-8950-72c6491500f6,AdmEvent::Node
8,60367d94-c71c-3936-9e50-2e0e7ff91f08,AdmFileObject::Node
9,3d5d883c-aa5d-39e8-a9d7-3e394da7657e,AdmFileObject::Node
10,a51e94d3-c36a-3881-8320-8215d5a2e006,AdmNetFlowObject::Node
11,c275bbbe-2964-3642-ac8b-751be17dfa3d,AdmNetFlowObject::Node
12,62945fa6-219d-37ad-9c55-6af0b6bf33b4,AdmEvent::Node
13,cafe8fbc-da2c-3479-8c63-dde615bd52bf,AdmNetFlowObject::Node
14,6ee95ea2-b661-3f43-b378-125dc13b3637,AdmNetFlowObject::Node
```

CHAPTER 4

PROJECT MANAGEMENT :

4.1 Project Management :

4.1.1 Collaborative Work :

4.1.1.1 Introduction :

To make this idea a reality and a success, we must all work together and cooperate in whatever way we can. Parallel jobs are also a terrific way to save time and money on projects, but we need proper tools to use them. As a result, we'll go over those tools as well as project monitoring in this chapter.

4.1.1.2 Tools :

- **Git:** Git and GitHub are popular solutions for managing various versions of data analytic information and allowing different people to collaborate on a project at the same time(Fiksel et al. 2019).
- **Google Drive:**Google Drive is a cloud storage service that, like many cloud services, is designed to relieve some of the strain on your hard drive. Cloud storage frees up space on your PC by transferring your data to its own remote servers, or "cloud." This frees up capacity on your devices for more vital tasks, such as downloading huge programs and games.
- **Google Meet :**Zoom is a video conferencing platform that allows people to connect online for video conference meetings, webinars, and live chat via a PC desktop or mobile app.
- **Google Colab:**We utilized Google Colab to share our artificial intelligence algorithms notebooks, synchronize our work and data from Google Drive, and communicate our findings.

4.1.1.3 Project Monitoring :

In terms of project monitoring, we spoke with the Classbox supervisor virtually every week and held meetings every 2 to 3 weeks to review progress. We had online sessions with

our university supervisor whenever we needed him, and he was always willing to help and give ideas and perspectives.

4.1.2 Project Management Methodology :

4.1.2.1 Introduction :

The usage of project management approaches has grown in popularity in recent years. Traditional options like PMP or Prince 2 are available, as well as agile approaches like SCRUM, Kanban, DSDM, and Extreme Programming.

Certification in one of these management approaches adds value to individuals and businesses alike, and it has even been used as a criterion for working with particular customers.

Why is it necessary to use project management methodologies? What are the advantages of operating in accordance with their recommendations? The primary goal of project man-

Figure 4.1: Project management methodology

agement technique is to standardize, structure, and coordinate work processes. This allows us to keep all initiatives on track and repeat successful parts while learning from failures, resulting in a continual improvement process. To put it another way, a technique is an excellent instrument for increasing efficiency when it is used.

A project management methodology is used to obtain certain benefits:

- Organizing project times.
- Providing tools to estimate times and costs correctly.
- Helping to manage and minimize project risks.
- Improving the cost-benefit relationship of resources.
- Developing the team's skills.

In terms of resources, a methodology may assist decrease the learning curve for a team, and it can be refined and altered as it is utilized in projects to match the company's particular style. It is feasible to reduce implementation risks and improve work by focusing on an appropriate and standardized approach.

4.1.3 Project management methodology:

Because management techniques are not all ideal for all projects, it is critical to understand their strengths so that they may be implemented effectively.

Figure 4.2: Project management methodology

4.1.4 Tasks :

The table 4.3 illustrates the different tasks that have been performed during the project realisation, with a timeline that represents the start and the end dates of each task:

ID	Name	Start Date	End Date	Duration	Progress %	Dependency
1	Master Thesis	janv 12, 2022	janv 19, 2022	6 days	100	
2	Trying to undrstand the topic	janv 12, 2022	janv 19, 2022	6 days	100	
3	First Meeting	janv 19, 2022	janv 19, 2022	1 day	100	2FS-1 days
4	Give us some articles to strat the master redaction	janv 20, 2022	janv 27, 2022	6 days	100	3FS
5	Stat Of art rediction	janv 28, 2022	févr 04, 2022	6 days	100	4FS
6	Master's bachground rediction	févr 07, 2022	févr 14, 2022	6 days	100	5FS
7	Master 's Conclusion +resume	févr 15, 2022	févr 22, 2022	6 days	100	6FS
8	Master 's introduction	févr 23, 2022	mars 02, 2022	6 days	100	7FS
9	master correction	mars 03, 2022	mars 10, 2022	6 days	100	8FS
10	Start PFE	mars 11, 2022	mars 18, 2022	6 days	100	9FS
11	Learn basic tools and Languages	mars 21, 2022	avr 04, 2022	11 days	100	10FS
12	Trying to be familiar with the data	avr 05, 2022	avr 12, 2022	6 days	100	11FS
13	Execute some existing model with our data	avr 13, 2022	avr 20, 2022	6 days	100	12FS
14	Implementation	avr 21, 2022	mai 23, 2022	23 days	100	13FS
15	Model's test	mai 16, 2022	juin 06, 2022	16 days	100	14FS-6 days
16	interface implémentation	juin 07, 2022	juin 20, 2022	10 days	100	15FS
17	Thesis rédaction	juin 07, 2022	juin 20, 2022	10 days	100	15FS

Figure 4.3: Tasks table that have been performed during this project.

Part III

Our Contribution

CHAPTER 5

ARTIFICIAL INTELLIGENCE ALGORITHMS

In this chapter, we will detail each AI algorithm that we have implemented in this project.

5.1 Artificial Intelligence Algorithms:

5.1.1 Rule Mining :

5.1.1.1 Itemset search

Let (R, O, A) be the formal context, where $O = \{o_1, o_2, \ldots, o_m\}$ is a set of objects, $A = \{a_1, a_2, \ldots, a_n\}$ is a set of attributes, *aka* items, and $R \subseteq O \times A$ is a binary relation such that $R(o, a)$ means object o has attribute a.

We define an item as a property of an object, and an itemset P, or a pattern, to be a set of items. An object $o \in O$ is said to include an itemset $P \subseteq A$, if $(o,p) \in R$, $\forall p \in P$. The number of items in an itemset determines the length of the itemset. The image of an itemset corresponds to the set of objects including the item (A. C. e. al. 2014; L. S. e. al. 2007).
Let f be a function that assigns for each itemset P, the list of all objects that include P: $f(P) = \{o \in O \mid o \text{ includes } P\}$. For instance, from Table 5.1, $f(e) = \{O_1\}$. We define the *support* of an itemset as the corresponding number of objects including the itemset, i.e. $supp(P) = |f(P)|$. We can also define the support as the relative ratio that corresponds to the proportion of objects including the itemset with respect to the whole population of objects in the database, i.e. $supp(P) = |f(P)|/|O|$. An itemset is said to be *frequent* if its support is greater than or equal to a given frequency threshold minimum support (denoted by *min_supp*). An itemset is said to be *rare* if its support is smaller than a given frequency threshold. An itemset G is said to be *generator* if it has no proper subset H ($H \subset G$) with the same support. Let FI denotes the set of frequent itemsets, RI the set of rare itemsets.

Objects / Items	a	b	c	d	e
O_1	0	1	1	0	1
O_2	1	0	1	1	0
O_3	1	1	1	1	0
O_4	1	0	0	1	0
O_5	1	1	1	1	0
O_6	1	0	1	1	0

Table 5.1: An example of a formal context, in which are represented a group of objects and their shared attributes.

An itemset X is said to be *closed* if there exists no proper superset Y ($X \subset Y$) with the same support, i.e. $CI = \{X | X \in \text{FI} \wedge \nexists Y \in \text{FI}$ such that $X \subset Y \wedge Supp(X) = Supp(Y)\}$.

The closure of an itemset X (denoted by $\gamma(X)$) is the largest superset of X with the same support. By consequence, if $\gamma(X) = X$ then X is a closed itemset. A frequent itemset is said to be *maximal frequent itemset* (MFI) if all of its supersets are rare. A rare itemset is said to be *minimal rare itemset* (MRI) if all of its subsets are frequent (Huang 2012). From the formal context in Table 5.1, and with *min_supp*=3, we have: {ac}: is a frequent itemset of length 2 and support 4, and {abcd}: is a rare itemset of length 4 and support 2. It can be noticed that the support is a monotonously decreasing function, with respect to the length of an itemset. When the number of attributes in the database is n, the number of itemsets is equal to 2^n. Thus, a direct search for the frequent itemsets by testing the itemsets that are frequent (or rare) is not conceivable. It is thus necessary to design heuristics, which avoid the exploration of the search space of all possible itemsets and that process each itemset as efficiently as possible. To that aim, many categories of itemsets classification algorithms have been proposed in the literature. Some of the most commonly used are *Apriori*, *FP-Growth*, *Eclat*, *H-Mine*, and *LCM* (Fournier 2017). All of these methods have the same input and the same output. However, they differ in the strategies and data structures they employ to discover itemsets efficiently. More specifically, the key differences could be either in whether they use a depth-first or breadth-first search, the type of database representation they use internally or externally, how they generate or determine the next itemsets to be explored in the search space, and how they count the support of itemsets to determine if they satisfy the minimum support constraint. For instance, in the Levelwise category (or Breadth-First), the algorithm explores the search space of itemsets by first considering 1-itemsets, then 2-itemsets, 3-itemsets, and lastly m-itemsets. *Apriori* (R. e. a. Agrawal 1994) is a well known levelwise algorithm for finding all frequent itemsets in a dataset using two properties (i) downward closure: All subsets of a frequent itemset are frequent, and (ii) anti-monotonicity: All supersets of a rare itemset are rare. Later, other variants of the *Apriori* algorithm have been introduced for optimizing the process of itemsets extraction, for instance *Apriori-Close*, *Pascal*, *Zart*, ... *etc* (N. e. a. Pasquier 1999). Contrarily to the levelwise algorithms, vertical methods use vertical representations of the database and a depth-first exploration to avoid saving many itemsets in memory. *Eclat* (M. J. Zaki 2000) and *Charm* (M. J. e. a. Zaki 2002) are the most well known vertical algorithms that use vertical layouts of the databases. They showed significant performance improvements over the level wise algorithms that usually use horizontal representations. .

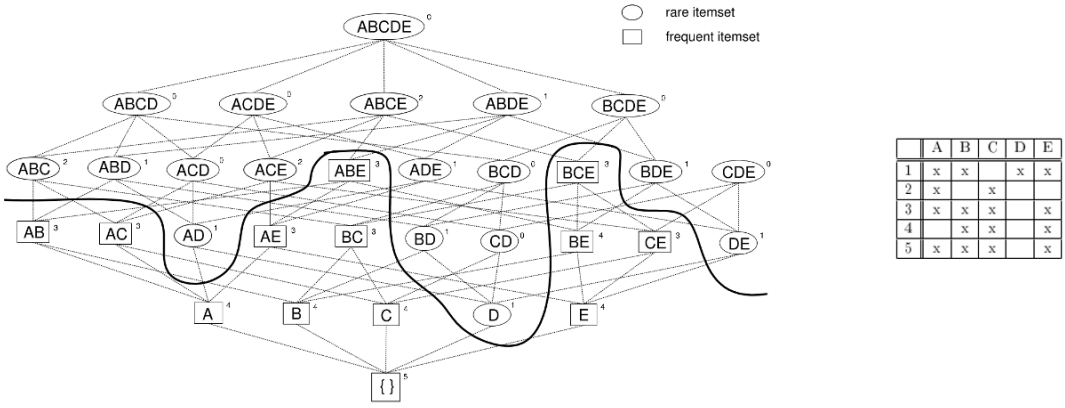
5.1.1.2 Association rule mining

In a complementary way to the previously described itemset patterns, association rule mining (ARM) was introduced by Agrawal et al. (R. Agrawal, Imieliński, and Swami 1993) with the main goal of discovering relationships, interesting correlations, valuable patterns, or associations among sets of items in a database. The relationships are not based on intrinsic properties of the data themselves but rather based on the co-occurrence of the items within the database.

An association rule has the form $A \rightarrow B$, where A and B are two itemsets and $A \cap B = \emptyset$. The left-hand side (LHS) A is referred to be the *antecedent* of the rule, and the right-hand side (RHS) B as the *consequent*. The quality of an association rule might be assessed through many interestingness measures. For instance, the *support* of the rule $A \rightarrow B$ is defined as the support of the itemsets $\text{Sup}(A \rightarrow B) = \text{Supp}(A \cup B)$, and the *confidence* of the rule is defined as the quotient $\text{Sup}(A \cup B)/\text{Supp(A)}$. The *confidence* can be seen as a conditional probability

P(B|A), i.e. probability of B knowing A. There exist many other interestingness measures that can be used for highlighting the quality of the association rules (Hahsler 2019). The *lift* or interest of the rule $A \rightarrow B$ is defined as Supp($A \cup B$)/Supp(A)×Supp(B). The *conviction* is defined as Supp(A) × Supp(¬B)/Supp(A ∪ ¬ B). The *dependency* is defined as |Supp(B|A)-Supp(B)| = |Supp($A \cup B$)/Supp(A)-Supp(B)|. The paradigm of association rule mining can be broadly classified into *frequent* and *rare* association rule mining based upon the type of knowledge derived from the databases. A rule is said to be *frequent* if its support is greater than or equal to a support threshold (min_supp). A rule is said to be *rare* if its support is smaller than a support threshold (max_supp). A rule is said to be *valid* (or strong) if its confidence is greater than or equal to a confidence threshold (denoted by min_conf).
For example, from the database in Table 5.1, and by considering min_supp=3, and min_conf= 3/5 we have: {ac} is a frequent itemset from which we can derive the rule $a \rightarrow c$ that is valid, with support= 4, and confidence=4/5. However, the itemset {abd} is not frequent (Supp=2), and consequently the rule $ab \rightarrow d$ is rare.VRARM (Valid Rare ARM) and VF-ARM (Valid Frequent ARM) are two independent anomaly identification algorithms that we developed. MRIs and MFIs, respectively, were used to construct the underlying association rules. The advantage of using both here is that it speeds up the ARM process compared to other direct ways since the underlying search space is much reduced. The pseudo-code of VR-ARM is presented in Algorithm 1. It requires a context (i.e., a mn-table) as well as support and confidence thresholds as input. It produces three outputs: I Rules, which are the association rules retrieved from C; (ii) Attacks, which is a list of anomalous items; and (iii) Scores, which is a list of scores connected with the objects in Attacks. The process begins by using GetRareRules to create rare rules from MRIs.It uses the Break the Barrier (BtB) approach. Following that, each object from C is compared to the rare associations: an object is said to meet a rare rule if the rule's itemsets are present in the object's itemsets. The degree of anomalousness of the matched object is represented by a score based on the set of matching outcomes (the higher the score, the more anomalous the object).Because VF-ARM is based on the same ideas as VR-ARM, it will be passed over here: The parameter min supp and a call to the MFI-based rule miner GetF reqRules are the most notable variations. Furthermore, if an object's itemsets are included in the rule's lefthand side but not its righthand side, the rule is broken. Objects that fit unusual rules or break common ones are classified as potentially anomalous and placed in Attacks. The average of a function combining the interestingness (e.g. lift) and length of the various rules, log2(1 Interest(R[j]) Length, is used to calculate their anomalousness scores (R[j]). High-quality rules with a lot of stuff on both sides would get a lot of points.

Finally, the prospective attack items are prioritized according to their anomaly scores, so that the top-ranked elements may be verified by a security expert.



	A	B	C	D	E
1	x	x		x	x
2	x		x		
3	x	x	x		x
4		x	x		x
5	x	x	x		x

Figure 5.1: Mining association rules for anomaly detection.

5.1.1.3 Description of the rule mining anomaly detection method

Our rule mining-based anomaly detection module implements two algorithms: *VR-ARM* for Very Rare Association Rule Mining and *VF-ARM* for Very Frequent Association Rule Mining-based anomaly detectors. The two types of association rules have been derived from the minimal closed rare itemsets ($MRIs$) and the maximal closed frequent itemsets ($MFIs$) respectively. The advantage behind using both $MRIs$ and $MFIs$ here is to make the process of generating the association rules very fast compared to other direct approaches (L. S. e. al. 2012), since the search space would be reduced (*iceberg* of the itemsets' lattice) compared to the use of the superset of all itemsets. Algorithm 1 gives a description of VR-ARM only, and does take as input a formal context C in which m objects (processes) and their n attributes (features) are represented as a binary matrix. As explained in the previous section, the contexts are extracted from the traces databases. The remaining parameters are threshold values of the support $Supp$ and confidence $Conf$ needed to derive the rules, and are explored using a grid search defined by two intervals of $Supp$ and $Conf$. The outputs of the algorithm are (i) $Rules$: a list that contains the association rules extracted from the context C using the two parameters $Supp$ and $Conf$; (ii) $Attacks$: a list of anomalous objects; and (iii) $Scores$: a list of scoring values associated to each anomalous object in $Attacks$. The algorithm starts extracting $Rules$ by calling $GetRareRules$, to generate the compressed rare rules through the $MRIs$, using the BtB (Break the Barrier) heuristics (L. S. e. al. 2012). After that, each object from the context C is matched against the association rules (present in $Rules$), and the algorithm checks out whether the current object satisfies a rare rule or not. We consider that an object satisfies a rare rule if its itemsets are included in the rare rule's itemsets. An anomaly score is assigned based on the satisfaction matching. This score represents the degree of anomalousness of the attack object. Higher is the score, more anomalous is the detected entity.
The second algorithm that implements VF-ARM is very similar to the previous one (not shown here), since they both have similar parameters and outputs. The main differences between VF-ARM and VR-ARM are the types of the rules since VF-ARM calls $GetFreqRules$ to generate the frequent rules using the $MFIs$, and at each step checks if the current object violates a frequent rule. We consider that an object violates a frequent rule if its itemsets are included the left-hand side of the rule and not in its right-hand side.
Objects that satisfy the rare rules or violate the frequent rules are considered as part of the anomalous entities and are added to $Attacks$, since they have either an uncommon behavior in the first case, or violate the common behavior in the second case. The violation score of each

attack object is calculated through the interestingness function (Confidence or Lift) of the of the *anomalous* association rule $R[j]$ as an average of $log_2(1 - Interest(R[j])) * Length(R[j])$. At a final stage, the list of the attack objects are ranked (descending order) towards their anomaly scores. Consequently the top ranked elements are the suspicious processes which would be checked later by the security expert.
The intervals used to define the domain of $Supp$ and $Conf$ include: Very Low values ($VL \leq 1\%$), Low values ($1\% < L \leq 30\%$), Average ($30\% < AV \leq 60\%$), High ($60\% < HI \leq 80\%$), and Very High ($80\% < VH \leq 100\%$).

Algorithm 1: VR-ARM: Rare Rule mining anomaly detection.

inputs : A formal context $C[m, n]$, $Supp$, $Conf$
outputs: $Rules[0 : r]$; // *A list of association rules*
$Attacks[0 : a]_{a \leq m}$; //*A list of anomalous objects*
$Scores[0 : a]_{a \leq m}$; // *A list of attack scores*
begin
  $Rules \longleftarrow$ GetRareRules(C,$Supp$,$Conf$);
  foreach *Object* $O[i]_{1 \leq i \leq m}$ *in* C **do**
    $Score[i] = 0.0$;
    $IsAttack = False$;
    foreach *Rule* $R[j]$ *in Rules* **do**
      if $O[i]$ *satisfies* $R[j]$ **then**
        $IsAttack = True$;
        $Score[i] = Score[i] + |log_2(1 - Interest(R[j])) * Length(R[j])|$
      end
    end
    if *IsAttack==True* **then**
      $Append(Attacks, O_i)$;
      $Append(Scores, Score[i])$;
    end
  end
  Rank(*Attacks* using *Scores*);
  return $(Rules, Attacks, Scores)$;
end

5.1.2 Autoencoders (AE):

Autoencoders (AE) are a type of neural network in which the input and output are the same. They are based on compressing the input into a latent-space representation and reconstructing the output from this representation.
In this part we just build plain-vanilla model with the follwing number of layers [25, 2, 2, 25] in which The input layer and the output layer has 25 neurons each. There are two hidden layers, each has two neurons. Concerning the training phase, we used the "relu" activation function in the hidden layers and the sigmoid activation function on the output. The model is compiled with Mean Squared Logarithmic loss and Adam optimizer. The model is then trained with 100 epochs with a batch size of 32. After building the model and setting it up, we were ready to fit the model on the training data and measure its performance via the use of confusion matrices.
For the AE model , we used dense layer and droupout layer for setting the outgoing edges of hidden units (neurons that make up hidden layers) to 0 at each update of the training phase to prevent the model from overfitting and the figure 5.2 shows the model summary and the

number of training parameters.

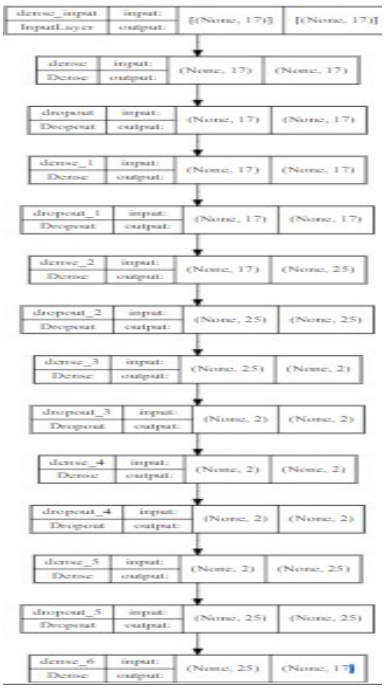


Figure 5.2: AE Model Architecture (summary)

In order to visualize the models and give a more details and in-depth view, we used Netron, and the resulting figure is as follow 5.3

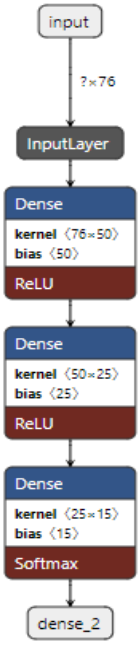


Figure 5.3: AE Architecture visualization

5.1.3 Isolation Forest (iForest):

Decision trees are used to create isolation forests. Because there are no pre-defined labels, they are implemented without supervision. Isolation forests were created with the assumption that anomalies in a dataset are "few and distinct" data points. only a few lines of code:

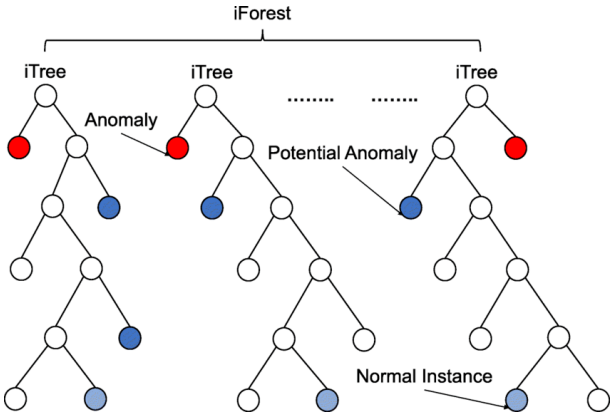


Figure 5.4: Isolation Forest (Regaya, Fadli, and Amira 2021)

We'll use Python to create isolation forest and see how it finds abnormalities in a dataset. We're all familiar with the fantastic scikit-learn API, which provides a variety of APIs for straightforward implementation. As a result, we'll use it to illustrate the usefulness of Isolation Forests for anomaly detection. (Khan 2022)

The mean anomaly score of the trees in the Isolation forest is used to calculate the anomaly score of an input sample. After fitting the full data to the model, the anomaly score is produced for each variable. When the anomaly score rises, it's more likely to be an oddity than a row with a lower anomaly value. This approach has three functions that make it simple to observe and save scores using only a few lines of code:

```
from sklearn.ensemble import IsolationForest
isolation_forest = IsolationForest(n_estimators=1000, contamination=0.08)
isolation_forest.fit(df['Rate'].values.reshape(-1, 1))
df['anomaly_score_rate'] = isolation_forest.decision_function(df['rate'].values.reshape(-1, 1))
df['outlier_univariate_rate'] = isolation_forest.predict(df['rate'].values.reshape(-1, 1))
```

5.1.4 One class SVM (OC-SVM):

SVM is also becoming more popular in one-class problems, in which all data belongs to the same class. In this example, the algorithm is taught what is "normal" so that when fresh data is presented, it can determine if it belongs in the group or not. If not, the new data is classified as unusual or anomalous.Alam 2020

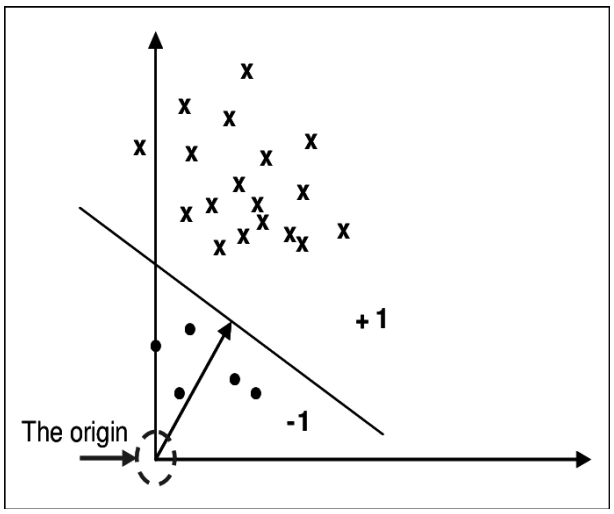


Figure 5.5: Classification in one-class SVM (Alashwal, deris, and Othman 2006)

This an example of the code we used:

```
from sklearn.svm import OneClassSVM
pred=clf.predict(X)
anomaly_score=clf.score_samples(X)
clf = OneClassSVM(gamma='auto',nu=0.04,gamma=0.0004).fit(X)
```

5.1.5 Local Outlier Factor (LOF) :

The Local Outlier Factor (LOF) is a density-based outlier identification technique that discovers outliers by computing the local deviation of a given data point. It's good for detecting outliers in datasets with an uneven distribution. The density between each data point and its neighboring ones is used to determine the outlier.

This technique may be used to do two sorts of detection. Outlier detection is unsupervised, whereas novelty detection is semi-supervised since it uses train data to make predictions on test data, despite the fact that train data does not provide accurate predictions.

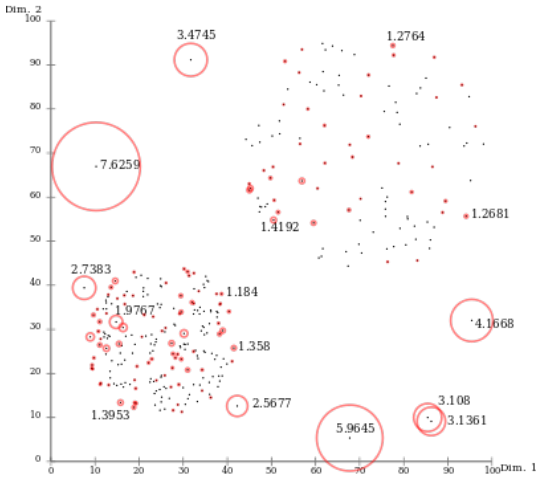


Figure 5.6: Visual Representation of Local Outlier Factor Scores where the black points are the Data ponits and the red cercles are the outlier Scores

Even though LocalOutlierFactor utilizes the same algorithm, it is only utilized for novelty detection in this model when the novelty option is True. Outlier detection would be employed

if the default is False, and just fit predict would function. This function is deactivated when novelty=True.

```
minmax = MinMaxScaler(feature_range=(0, 1))
X = minmax.fit_transform(df[['rate','scores']])
# Novelty detection
clf = LocalOutlierFactor(n_neighbors=100, contamination=0.01,novelty=True) #when novelty = True
clf.fit(X_train)
df['multivariate_anomaly_score'] = clf.decision_function(X_test)
df['multivariate_outlier'] = clf.predict(X_test)
# Outlier detection
local_outlier_factor_multi=LocalOutlierFactor(n_neighbors=15,contamination=0.20,n_jobs=-1) # whe
multi_pred=local_outlier_factor_multi.fit_predict(X)
df1['Multivariate_pred']=multi_pred
```

5.1.6 Elliptic Envelope Algorithm:

When the data is Gaussian distributed, this algorithm is utilized. This model turns the data into an elliptical shape, and the points that are far away from the shape coordinates are regarded outliers, and the minimum-covariance-determinant is determined as a result. It's similar to determining the covariance in a dataset so that the lowest value is discarded and the highest values are deemed anomalies. The rationale for this algorithm detection is depicted in the diagram below (PEDDIREDDI 2021).

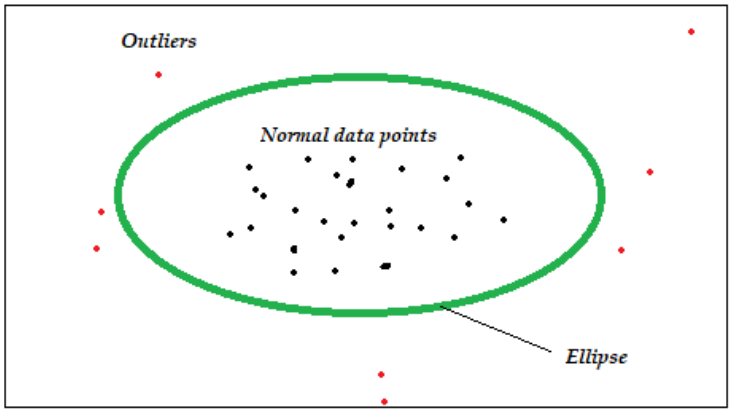


Figure 5.7: Elliptic Envelope

It uses the same line of code to fit the data and anticipate what will happen next, identifying anomalies in the data with a -1 for anomalies and a +1 for normal data or in-liers.

persistent if this approach of storing and recovery is used.

```
from sklearn.covariance import EllipticEnvelope
model1 = EllipticEnvelope(contamination = 0.1) # fit model
model1.fit(X_train)
model1.predict(X_test)
```

5.1.7 Meta Learning:

The concept of meta learning refers to the utilization of several classification algorithms in a combined mode, in-order to enhance the accuracy and the efficiency of the model. In our case we combined the following algorithms (Rule Mining ,IForest , OC-svm, AutoEncoders, LOF , EllipticEnvlope) to maximise the classification rates of the true positives.the results shaws in the table 5.9

5.2 Results and Discussion :

5.2.1 Datasets:

For our evaluation, we used two data collections described in (Berrada et al. 2020), publicly available [1]. These collections (or scenarios) consist of sets of feature views or contexts from raw whole-system provenance graphs produced during two DARPA Transparent Computing (TC) program "engagements" (exercises aimed at evaluating provenance recorders and techniques to detect APT activity from provenance data). During the engagements, several days (5 for scenario 1 and 8 for scenario 2) of system activity (process, netflow, etc.) were recorded on various platforms (Windows, BSD, Linux and Android) subjected to APT-like attacks.(Han, T. Pasquier, and Seltzer 2018) explains engagements in more detail and provides more information on provenance data for Scenario 2 (referred to as Engagement 3). All contexts in the data collections relate unique process identifiers (rows) to varying features/attributes of process activity: types of events performed by process (ProcessEvent (PE)), name of process executable (ProcessExec (PX)), name of parent process executable (ProcessParent (PP)), IP addresses and ports accessed by process (ProcessNetflow (PN)), joining all the previous features (renamed, if needed, to avoid ambiguity) together (ProcessAll (PA)). Table 5.2 summarizes the properties of all contexts per collection/scenario. For a more detailed description, see (Berrada et al. 2020). We might observe that (a): The number of processes varies widely: Linux has 3 to 10-times more that Windows/BSD and up to 2400 times more than Android. (b): The number of processes/attributes is unrelated to the original dataset size. Thus, albeit the largest, the Android dataset has the fewest processes and attributes (due to the provenance recorder mostly logging low-level app activity and performing dynamic information flow tracking, which are pieces of information we do not analyze).

	Scenario	Size	*PE*	*PX*	*PP*	*PN*	*PA*	*nb_attacks*	$\%\frac{nb\_attacks}{nb\_processes}$
BSD	1	288 MB	76903 / 29	76698 / 107	76455 / 24	31 / 136	76903 / 296	13	0.02
	2	1.27 GB	224624 / 31	224246 / 135	223780 / 37	42888 / 62	224624 / 265	11	0.004
Windows	1	743 MB	17569 / 22	17552 / 215	14007 / 77	92 / 13963	17569 / 14431	8	0.04
	2	9.53 GB	11151 / 30	11077 / 388	10922 / 84	329 / 125	11151 / 606	8	0.07
Linux	1	2858 MB	247160 / 24	186726 / 154	173211 / 40	3125 / 81	247160 / 299	25	0.01
	2	25.9 GB	282087 / 25	271088 / 140	263730 / 45	6589 / 6225	282104 / 6435	46	0.01
Android	1	2688 MB	102 / 21	102 / 42	0 / 0	8 / 17	102 / 80	9	8.8
	2	10.9 GB	12106 / 27	12106 / 44	24 / 11	4550 / 213	12106 / 295	13	0.10

Table 5.2: Experimental dataset metrics. A context entry (columns 4 to 8) is number of rows (processes) / number of columns (attributes).

The Figure 5.8 shows an overview of the dataset :

[1] https://gitlab.com/adaptdata/e2/-/tree/master/pandex

	Object_ID	X0_EVENT_CLOSE	X0_EVENT_READ	X0_EVENT_WRITE	X0_EVENT_OPEN	X0_EVENT_SIGNAL	X0_EVENT_SENDTO	X0_EVENT_LSEEK	X0_EVENT_ACCEPT	X0_EVENT_FORK	...	X3_53	X3_50387	X3_57125	X3_59953	X3_57348
0	2b8d322e-a990-3d31-b5e4-17a591878388	1	1	0	1	1	1	1	0	1	...	0	0	0	0	0
1	ffbc20e6-7201-3171-ba73-9fd769f462f1	1	1	0	1	0	0	1	0	1	...	0	0	0	0	0
2	a12db087-412a-3fb2-a5c2-65cfb8cf3dd2	1	1	0	1	0	1	1	0	1	...	0	0	0	0	0
3	1342a735-ce6c-3534-a879-445bbfee6119	1	1	0	1	0	0	1	0	1	...	0	0	0	0	0
4	7a215537-a242-3767-a025-7490fdee1fde	1	1	0	0	0	0	0	0	1	...	0	0	0	0	0
5	9a9db827-3306-31dd-b51d-9a51d9a2d69a	0	0	0	0	0	0	0	0	1	...	0	0	0	0	0

Figure 5.8: Process Event_e2 Data description

The Figure 5.9 shows shows the dotted "synthetic" edges, and the context menu defined by the UI language making high-level semantic queries easily accessible to the analyst :

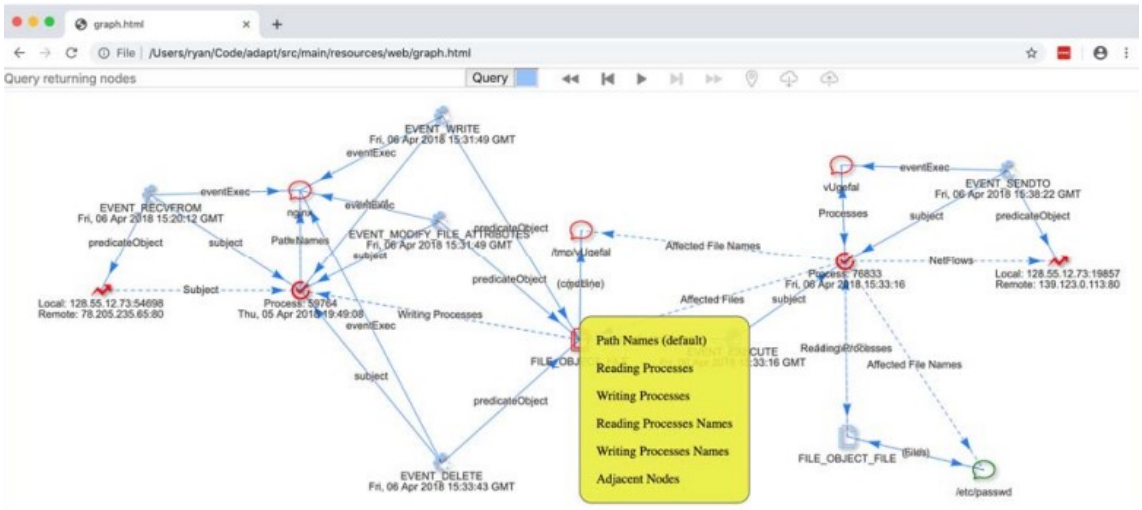


Figure 5.9: Example Screenshot of the Exploration and Explanation UI

5.2.2 Evaluation Metrics :

Rather than categorizing processes as anomalous or not, our system rates them according to how suspicious they are. The accuracy paradox would result from utilizing classification with accuracy due to the huge data imbalance (attacks range from 0.004 percent to 8.8 percent of data) (Thomas and Balakrishnan 2008).

Instead, normalized discounted gain (nDCG), which was initially suggested in (Järvelin and Kekäläinen 2002) as a way to quantify ranking quality in information retrieval, might be a better fit. It is designed to represent the value a ranking item is assigned by the end user, i.e., relevant materials are more valuable than moderately relevant documents, and even more valuable than irrelevant documents. Furthermore, because a user would only read a limited amount of a ranking, relevant papers towards the top of the list have more value than those farther down the list (those too far down would be skipped). As a result, nDCG prefers rankings that include all important papers towards the top.

The same idea applies to anomalous processes: low-ranking attack processes are nearly worthless to analysts, whose monitoring burden grows in tandem with the number of events

to examine (until suspicious processes escape their notice). A huge number of top-ranked regular processes (false alarms) may also undermine analyst trust in the automated monitoring system. A ranking's discounted cumulative gain (DCG) is calculated by adding element relevance ratings that have been penalized by the logarithm of their rank:

$$DCG_N = \Sigma_{i=1}^{N} \frac{rel_i}{log_2(i+1)} \tag{5.1}$$

where N is the ranking size and rel_i is the relevance score of the i-th element. Because DCG scores for lists of varied sizes are not comparable, a normalization is performed using the ideal score iDCG, i.e. one that corresponds to the optimal scenario (all relevant entities at the same level).Thus, for a ranking with p relevant entities

$$IDCG_p = \Sigma_{i=1}^{REL_p} \frac{rel_i}{log_2(i+1)} \tag{5.2}$$

The normalized form of DCG is nDCG:

$$nDCG_p = \frac{DCG_p}{IDCG_p} \tag{5.3}$$

5.3 Evaluation Results:

5.3.1 APT Ranking Visualisation with Band Diagrams :

We developed a visualization approach called band diagram charts to make inspecting the produced ranks easier (see figure 5.10). Each sorted list is shown by a horizontal band, with red lines indicating the location of a genuine positive (APT). Top-ranked entities will be placed on the left, revealing a highly efficient anomaly detection system with several red lines in that region.

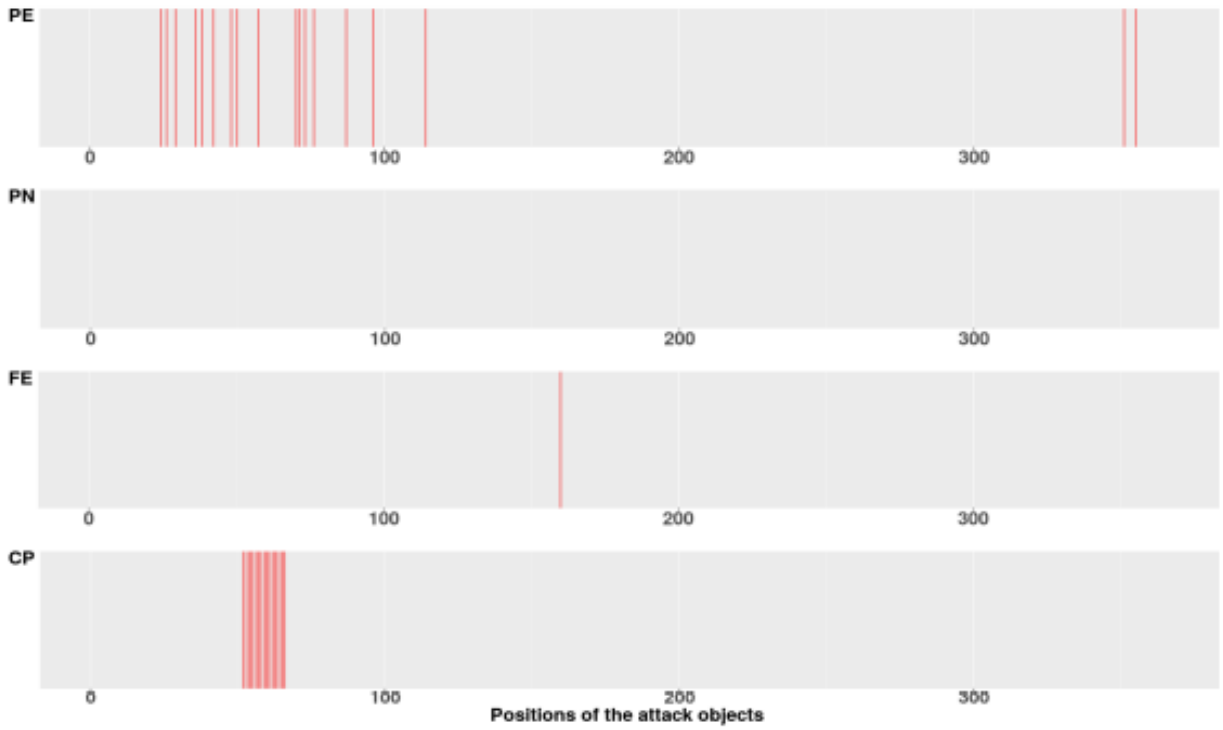


Figure 5.10: Band diagrams representing the positions of the attacks (true positives) in some contexts of the BSD dataset (scenario 1). The x-axis represents the attack positions in ranked lists.

5.3.2 Ranking Evaluation With nDCG:

We compared the different algorithms that we developed by retrieving their produced nDCG scores, namely Auto Encoders, Local Outlier Factor (LOF), Iforst , One Class SVM ,VF-ARM, VR-ARM and Elliptic Envelope. We ran them on the contexts that are presented in the Table 6.2 and assessed the resulting rankings by means of nDCG. To that end, we first implemented LOF, OcSVM,VF-ARM and VR-ARM, Iforst , and Elliptic Envelope in Python.

Tables 5.3 to 5.7 illustrate the average of nDCG scores into each context for a given operating system. A combination is represented by an entry here (method, scenario, OS). Notably, certain algorithms did not complete in an acceptable amount of time (4 to 48 hours). DNF flags these kind of situations. The maximum nDCG score for each OS (row) versus scenario combination is shown in red (orange represent the second max value in each row).

Autoencoders and Rule mining were competitive on the ProcessEvent context with considerably better scores obtained with AutouEncoders around 0.92 (Android, scenario 1) and 0.88 (Android, scenario 1). Ocsvm and iforest yielded similar scores to each other for the different databases. Iforest, AE and rule mining produced good scores with(Android, scenario 1) concerning ProcessExec and ProcessParent and ProcessNetflow. AE and Rule mining generated moderately acceptable scores with all four OSes. The highest scores in these contexts were obtained by AE (0.98) in BSD w.r.t. scenario 1 and 2. LOF, Elliptic and OCsvm generated poor scores for all four OSes and contexts compared with the other methods.

Table 5.8 summarizes the detection and classification of the APT attacks for each operating system and scenario group, highlighting the winner method based on ndcg scores. We

can observe that our method that is based on the auto-encoders outperforms the remaining approaches with nDCG scores varying between 0.87 up to 0.97 which is highly acceptable.

	Detection Method	OC-SVM		I-Forest		LOF		AE		VR-ARM		VF-ARM	
	Attack Scenario	1	2	1	2	1	2	1	2	1	2	1	2
Source	Windows	0.14	0.16	0.14	0.16	0.19	0.16	0.87	0.23	0.82	0.19	0.33	0.13
	BSD	0.14	0.12	0.13	0.12	0.16	0.16	0.93	0.96	0.64	0.12	0.33	0.12
	LINUX	0.17	0.18	0.17	0.18	0.17	0.17	0.96	0.96	0.13	0.14	0.22	0.10
	Android	0.44	0.18	0.32	0.18	0	0.33	0.92	0.77	0.87	0.40	0.77	0.12

Table 5.3: Evaluation of anomaly scoring: ProcessEvent

	Detection Method	OC-SVM		I-Forest		LOF		AE		VR-ARM		VF-ARM	
	Attack Scenario	1	2	1	2	1	2	1	2	1	2	1	2
Source	Windows	0.17	0.15	0.37	0.27	0.19	0.16	0.84	0.91	0	0	0	0
	BSD	0.14	0.13	0.58	0.22	0	0.16	0.88	0.89	0.08	0.05	0.53	0.06
	LINUX	0.18	0.17	0.27	0.33	0.19	0	0.89	0.96	0.12	0	0.10	0.004
	Android	0.37	0.17	0.78	0.49	0	0.33	0.95	0.45	0	0	0	0

Table 5.4: Evaluation of anomaly scoring: ProcessExec

	Detection Method	OC-SVM		I-Forest		LOF		AE		VR-ARM		VF-ARM	
	Attack Scenario	1	2	1	2	1	2	1	2	1	2	1	2
Source	Windows	0.09	0.30	0.15	0.17	0.10	0.16	0.68	0.23	0	0	0	0
	BSD	0.11	0.10	0.11	0.10	0	0.17	0.97	0.88	0.29	0.24	0.67	0.06
	LINUX	0.21	0.19	0.17	0.32	0.17	0.17	0.84	0.90	0	0	0	0
	Android	DNF	0	DNF	0	0	DNF	0.91	0.91	0	0	0.12	0.03

Table 5.5: Evaluation of anomaly scoring: ProcessParent

	Detection Method	OC-SVM		I-Forest		LOF		AE		VR-ARM		VF-ARM	
	Attack Scenario	1	2	1	2	1	2	1	2	1	2	1	2
Source	Windows	0.32	0.11	0.33	0.11	0.28	0.11	0.68	0.47	0.63	0.30	0	0
	BSD	0.10	0.12	0.25	0.12	0.11	0.19	0.97	0.80	0.34	0.60	0.11	0.18
	LINUX	0.32	0.36	0.21	0.20	0.25	0	0.84	0.46	0.58	0.39	0.42	0.11
	Android	0.50	0.16	0.68	0.23	0	0	0.91	0.43	0.46	0.71	0.12	0.03

Table 5.6: Evaluation of anomaly scoring: ProcessNetflow

	Detection Method	OC-SVM		I-Forest		LOF		AE		VR-ARM		VF-ARM	
	Attack Scenario	1	2	1	2	1	2	1	2	1	2	1	2
Source	Windows	DNF	DNF	DNF	DNF	0.16	DNF	DNF	DNF	0.61	0.35	0.50	0.07
	BSD	0.13	DNF	0.13	DNF	0	0	0.85	DNF	0.36	0.52	0.18	0.14
	LINUX	0.17	DNF	0.21	DNF	0.19	0	DNF	DNF	0.54	0.45	0.13	0.09
	Android	0.32	0.17	0.34	0.22	0	0.20	0.91	0.79	0	0.51	0	0.43

Table 5.7: Evaluation of anomaly scoring: ProcessAll

OS	Scenario 1			Scenario 2		
	Winner method	Context	Ndcg scores	Winner method	Context	Ndcg scores
Windows	Autoencoder	ProcessEvent	0.87	Autoencoder	Prosecc Exec	0.91
BSD	Autoencoder	ProcessParent / ProcessNetflow	0.97	Autoencoder	ProcessEvent	0.96
Linux	Autoencoder	ProcessEvent	0.96	Autoencoder	ProcessEvent	0.96
Android	Autoencoder	ProcessExec	0.95	Autoencoder	ProcessParent	0.91

Table 5.8: Winner methods over the databases / OSes.

	Detection Method	PE		PX		PP		PN		PA	
	Attack Scenario	1	2	1	2	1	2	1	2	1	2
Source	Windows	0.65	0.24	0.34	0.34	0.21	0.55	DNF	0.42	DNF	DNF
	BSD	0.51	0.18	0.51	0.44	0.49	0.31	0.44	0.56	0.53	DNF
	LINUX	0.26	0.28	0.32	0	0.32	0.39	0.58	0	DNF	DNF
	Android	0.90	0.86	0.89	0.62	0	0.91	0.89	DNF	0.89	0.51

Table 5.9: Evaluation of anomaly scoring: Meta Learning case

Part IV

Solution Design And Technologies

CHAPTER 6

SOLUTION DESIGN AND TECHNOLOGIES

6.1 Introduction :

In this chapter we are going to cover up the process of designing our web platform and the technologies that we are going to use to release the web application.

6.2 Conception and system design :

6.2.1 System Architecture :

The main modules in our system that we developed in this project are:

- **The Anomaly detection module:** The anomaly detection module is the core subsystem responsible of the following sub-tasks:
  - Loading the database (CVS files, or JSON)
  - Identifying the outliers.
  - Displaying the the results.
- **The GUI module :**This module is simply an intuitive and easy-to-use interface that allows the user to use the different functions provided by the previously mentioned module through the use of interactive web pages.

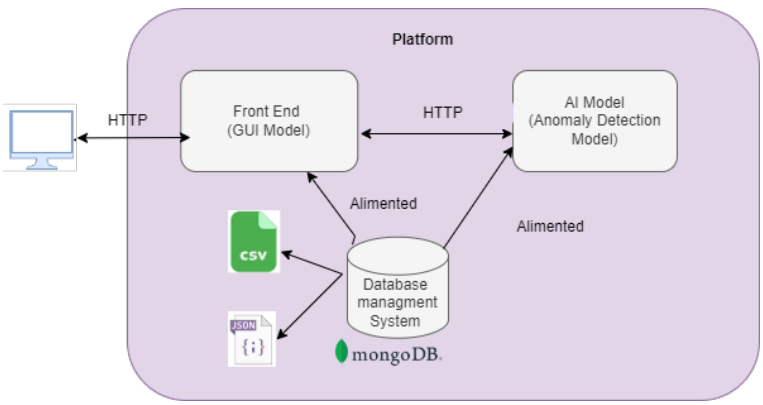


Figure 6.1: Global architecture of our system.

6.2.2 Modeling the requirements:

In order to provide more specifications and visualization on the artifacts of our system, this section will provide a global view of our application through the use of the different Unified Modeling Language (UML) diagrams [1].

6.2.3 Use case diagram:

A use case diagram in UML is a visual representation of a user's interaction with a system that shows the relationship between him and the many use case situations. For instance, the Figure 6.2 illustrates the use case of our model. Firstly, the user loads the database (here csv files). The second function would be running the model for identifying the outliers. The third function would be displaying the visualising the results.

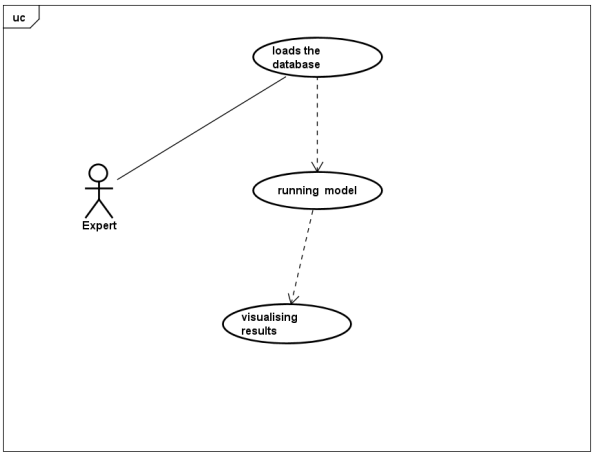


Figure 6.2: The use case diagram

6.2.4 Sequence Diagrams :

In this case scenario, the user can execute all the anomaly detection functionalities. The first function that can be done is to choose the algorithm he wants to work with and also the OS. In our case we have four OSes (Windows , Linux , Unix and Android). After selecting a given OS, the user should then select which process file he wants to test (the file that contains a group of object IDs and the activities they have performed 6.4) and also a validation database file 6.5. Figure 6.3 shows the sequence diagram with all the functionalities

[1] https://en.wikipedia.org/wiki/Use_case_diagram

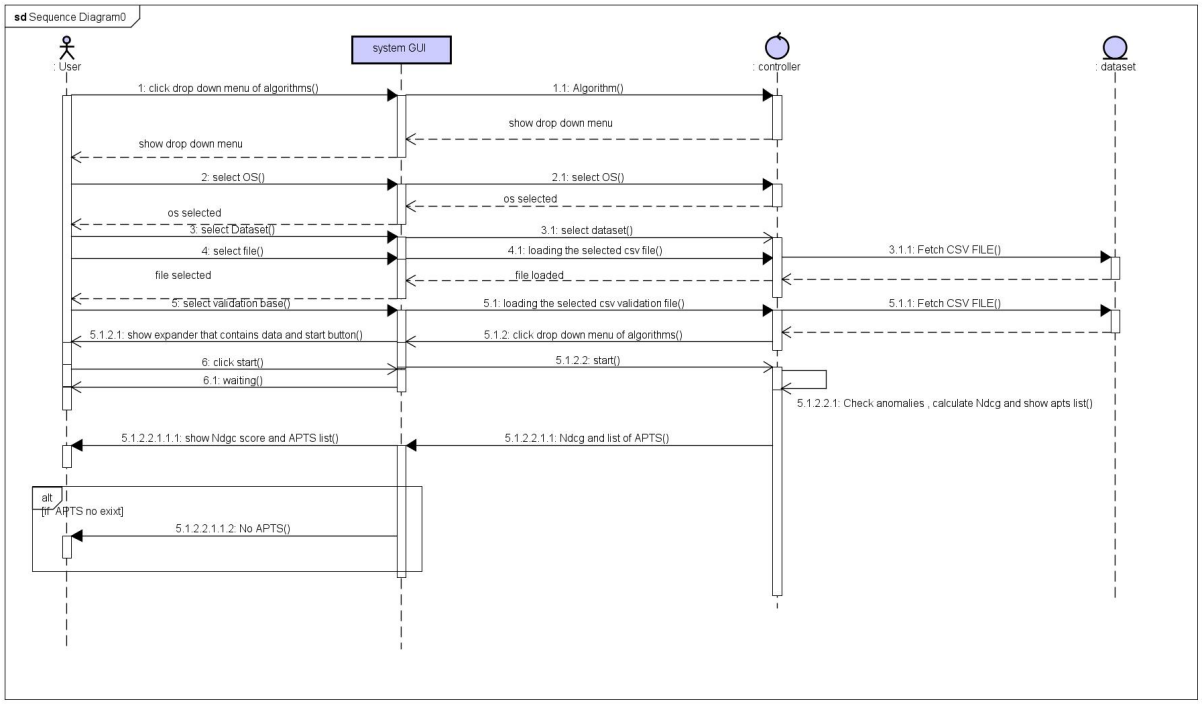


Figure 6.3: The Sequence Diagrams

	Object_ID	\win	\win	\win	\win	\win	\win	\win	\win	\
0	fivedirections-e3_d90ec938-a31c-38b1-bec	0	0	0	0	0	0	0	0	
1	fivedirections-e3_233b855f-d2b2-3cda-bbc	0	0	0	0	0	0	0	0	
2	fivedirections-e3_78b1873b-f28a-39e7-8ee	0	0	0	0	0	0	0	0	
3	fivedirections-e3_70f3f086-8ac5-35dd-bc7e	0	0	0	0	0	0	0	0	
4	fivedirections-e3_f7a7d331-e763-37a8-bbd	0	0	0	0	0	0	0	0	

Figure 6.4: Process Event from Windows OS. The rows here represent the uuids of the OS processes (programs), and the columns represent the actions that have been performed by each process.

uuid	label
fivedirections-e3_3d1e7ea5-f354-35cf-9db9-bd3459a14305	AdmSubject::Node
fivedirections-e3_263ca4ce-4263-3272-ba30-e0c33eed29f9	AdmSubject::Node
fivedirections-e3_d7a5b8a1-9fef-3b4a-8e85-1084e4065034	AdmSubject::Node
fivedirections-e3_9b229ce6-b07c-3574-84c1-7a7a06512775	AdmSubject::Node
fivedirections-e3_35197bcc-a159-3c6b-9b30-0b1bababb9522	AdmSubject::Node

Figure 6.5: Validation base

6.2.4.1 Graphical User Interface (GUI):

In this illustrative part, we provide some screenshots from the system's GUI for more understand the previous part .

This picture spotlights the homepage which encompasses a Navigation bar, a dropdown list that contains the algorithms, a set of radio buttons for each OS, Dataset files and a check box for the validation base.

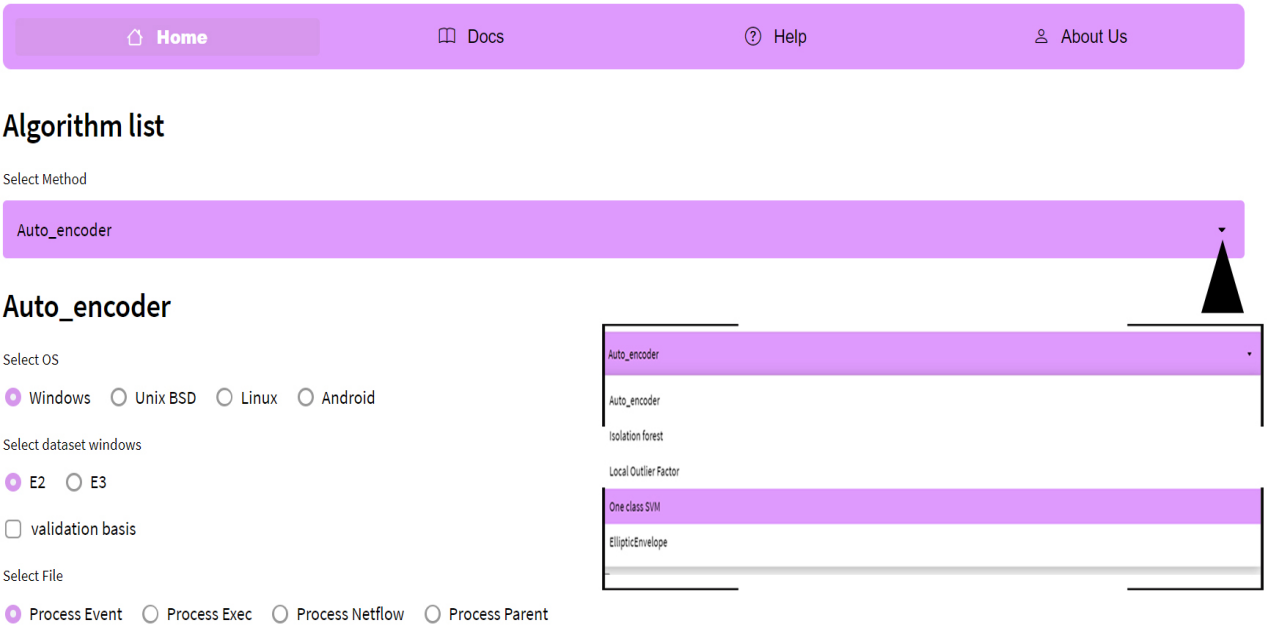


Figure 6.6: Home Page of the web interface of our APTs detection system.

When the user checks the validation base checkbox, an expander component of data shows up with a star button.

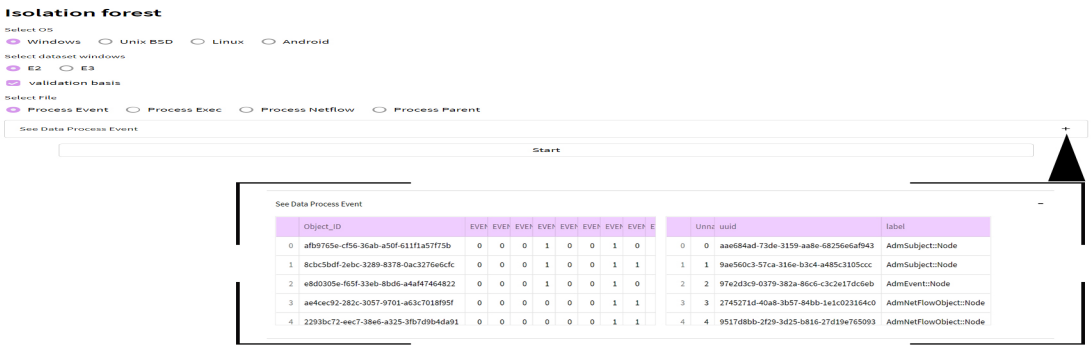


Figure 6.7: The expander object shows data and start button.

After clicking on the start button the algorithm module begins to run, and the results are displayed on the interface:

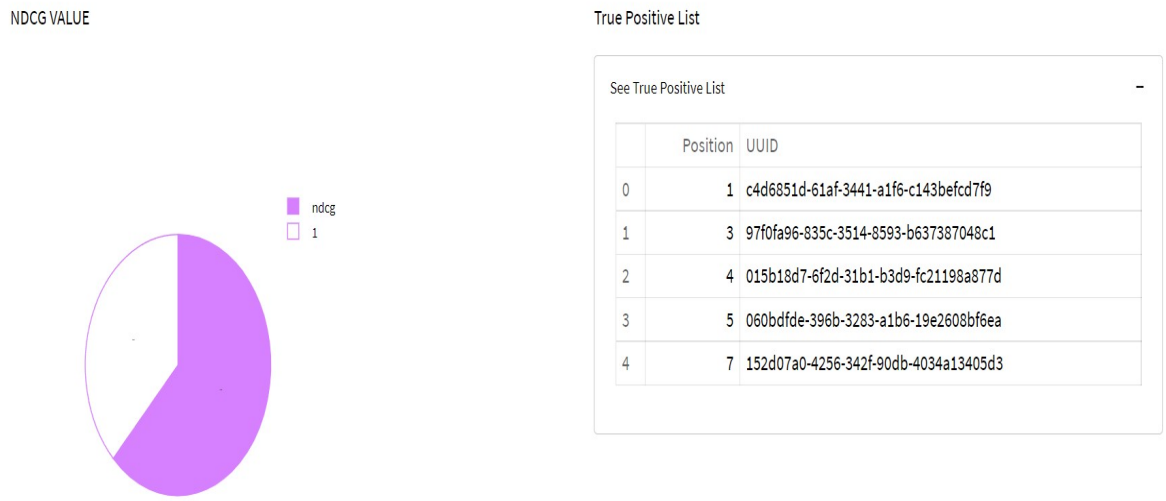


Figure 6.8: Illustration of some results : on the right hand side, we have got some True Positives (detected APTs), and on the left hand side we have the anomaly scores (Ndcg scores).

6.2.5 System Implementation Tools:

In this last part we are going to mention the tools that we used during our project implementation

6.2.5.1 Artificial Intelligence Technologies :

We'll go through all the technologies we have used in our implementation for both web and AI modules:

- Environnement :
  - **Python :** Guido Van Rossum designed the Python programming language in 1989. It's a programming language for interpreters that was created as an open source effort. Object-oriented programming, procedural programming, and functional programming are all supported by Python. Python is a cross-platform language, which means that programs developed in it run on a variety of operating systems, including Microsoft Windows, Linux, and Unix systems like Mac OS X, with nearly complete support for standard and third-party libraries, just by copying the program's source code(Mészárosová 2015).
- Libraries :
  - **Pandas :**Pandas is a Python module that allows you to work with large data collections. It offers tools for data analysis, cleansing, exploration, and manipulation. Wes McKinney came up with the moniker "Pandas" in 2008, which refers to both "Panel Data" and "Python Data Analysis."[2]
  - **Scikit-learn:**Scikit-learn is an open source Python toolkit that uses a uniform interface to construct a variety of machine learning, preprocessing, cross-validation, and visualization methods.

[2]https://www.w3schools.com/python/pandas/pandas_intro.asp

- **Matplotlib :**Matplotlib is a data visualization and graphical charting package for Python and its numerical extension NumPy that runs on all platforms. As a result, it provides an open source alternative to MATLAB. Matplotlib's APIs (Application Programming Interfaces) may also be used to incorporate charts in graphical user interfaces [3].
- **Imblearn:**Imblearn approaches are strategies for generating a data collection with an equal proportion of classes. This sort of data collection would allow the prediction model to generalize effectively [4].
- **Numpy :**NumPy is the most important Python module for scientific computing. It's a Python library that includes a multidimensional array object, derived objects (such as masked arrays and matrices), and a variety of routines for performing fast array operations, such as mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation, and more(NumPy community 2022).
- **Keras :** Keras is a deep learning platform that contains several routines. It is one the most well known DL library that can be run on different OSes. It is actually written with Python but has several interfaces with other programming languages.

- Tools:
  - **Anaconda :**Anaconda is a data science-focused open-source version of the Python and R programming languages that seeks to make package management and deployment easier. Conda, the package management system in Anaconda, manages package versions by analyzing the existing environment before completing an installation to prevent interrupting other frameworks and packages(team 2022).
  - **Google Colab :**Colab is a cloud-based Jupyter notebook environment that is free to use. Most importantly, it doesn't require any setup, and the notebooks you create may be modified concurrently by your team members, much like Google Docs pages. Many common machine learning libraries are supported by Colab and can be quickly loaded into your notebook(Point 2019).

6.2.5.2 Web Platform Technologies :

- **Streamlit:**

  Streamlit is a Python open source library that was released in 2018. The term "open source" refers to the fact that the source code is available to everyone. This means that it may be reused by all users to construct their own program, providing significant flexibility for everyone's demands. Open source, which has become a genuine movement in software development, promotes collaborative production and thereby increases software quality. [5]

- **Components :**create media pipelines. Media pipelines are sets of connected components with methods that allow you to control the pipeline and quickly add/remove

[3]https://www.activestate.com/resources/quick-reads/what-is-matplotlib-in-python-how-to-use-it-for-plotting/

[4]https://analyticsindiamag.com/what-is-imblearn-technique-everything-to-know-for-class-imbalance-issues-in-machine-learning/

[5]https://datascientest.com/streamlit-ou-loutil-pour-presenter-votre-travail-de-machine-learning

components, while components are a low-level abstraction on top of Node streams to allow two-way communication. Components are third-party modules that enhance the capabilities of Streamlit [6]

[6] https://streamlit.io/components

Part V

Conclusion and future works

CHAPTER 7

CONCLUSION AND FUTURE WORKS

7.1 Conclusion:

Many vital computer systems are vulnerable to a variety of assaults nowadays. These attacks can have catastrophic effects for both individuals and the environment in most circumstances.

This capstone project is meant to be a little piece of a larger puzzle that can help in the prevention of APT cyber assaults by assisting experts in monitoring their activities and offering comprehensive behavioral analysis to detect abnormalities and weaknesses.

In our thesis, we began the design, development, and testing of our APTs anomaly detection system, with the first chapter containing a requirements analysis, system architecture, and several UML models that define the functionality supplied by our application. We also shared some code and GUI samples from the program, as well as a breakdown of the implementation phases and the tools we utilized. We assessed the performance of our model and compared it to other outliers detection systems. We also emphasized the challenges and bottlenecks we faced, as well as our contribution to the project.

7.2 Future works:

Our awareness of the importance of maintaining APTs security as well as the application of machine learning for anomaly detection and the best practices related both network and software engineering have been deepened thanks to this final project. Our little job is only getting started and doesn't end here. This insight opens the door to bigger accomplishments in this field. Having stated that, the following views are in our minds for potential future works:

- First, by utilizing more recent datasets, we want to improve the generalization performance of our system to identify different types of assaults.
- Another crucial feature that may be added to our system is the ability to find anomalies in encrypted data.
- Finally, the use of a hybrid approach that involves the use of different machine learning algorithms.

BIBLIOGRAPHY

[Abd+19] Rusul Abduljabbar et al. "Applications of Artificial Intelligence in Transport: An Overview." In: *Sustainability* 11.1 (2019). ISSN: 2071-1050. DOI: 10.3390/su11010189. URL: https://www.mdpi.com/2071-1050/11/1/189.

[AIS93] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. "Mining Association Rules Between Sets of Items in Large Databases." In: *SIGMOD Rec.* 22.2 (June 1993), pp. 207–216. ISSN: 0163-5808. DOI: 10.1145/170036.170072. URL: http://doi.acm.org/10.1145/170036.170072.

[Agr94] Rakesh et al. Agrawal. "Fast Algorithms for Mining Association Rules in Large Databases." In: *Proceedings of the 20th Int Conf on Very Large Data Bases*. VLDB '94. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994, pp. 487–499. ISBN: 1-55860-153-8. URL: http://dl.acm.org/citation.cfm?id=645920.672836.

[al14] Aggarwal Charu et al. *Frequent Pattern Mining*. Springer Publishing Company, Inc., 2014. ISBN: 3319078208, 9783319078205.

[al07] Laszlo Szathmary et al. "ZART: A Multifunctional Itemset Mining Algorithm." In: *5th Int Conf on Concept Lattices and Their Apps, CLA, Montpellier, France, Oct 24-26*. 2007. URL: http://ceur-ws.org/Vol-331/Szathmary.pdf.

[al12] Laszlo Szathmary et al. "Efficient Vertical Mining of Minimal Rare Itemsets." In: *Proceedings of The Ninth International Conference on Concept Lattices and Their Applications, Fuengirola (Málaga), Spain, October 11-14, 2012*. 2012, pp. 269–280. URL: http://ceur-ws.org/Vol-972/paper23.pdf.

[Ala20] Mahbubul Alam. *Support Vector Machine (SVM) for Anomaly Detection*. https://towardsdatascience.com/support-vector-machine-svm-for-anomaly-detection-73a8d676c331. Oct. 2020.

[AdO06] Hany Alashwal, Safaai bin deris, and Razib Othman. "One-Class Support Vector Machines for Protein Protein Interactions Prediction." In: *Int J Biomed Sci* 1 (Jan. 2006).

[BML18] Sidahmed Benabderrahmane, Nedra Mellouli, and Myriam Lamolle. "On the predictive analysis of behavioral massive job data using embedded clustering and deep recurrent neural networks." In: *Knowledge-based systems* 151 (2018), pp. 95–113.

[Ber+20] Ghita Berrada et al. “A baseline for unsupervised advanced persistent threat detection in system-level provenance.” In: *Future Generation Computer Systems* 108 (2020), pp. 401–413.

[Boe22] Gaudenz Boesch. *The Most Valuable Computer Vision Smart City Applications (2022 Guide)*. https://viso.ai/applications/computer-vision-in-smart-city-applications/. 2022.

[Bon19] Anne Bonner. “The complete beginner’s guide to deep learning: Convolutional neural networks.” In: *Medium, towards data science* (2019).

[Fik+19] Jacob Fiksel et al. “Using github classroom to teach statistics.” In: *Journal of Statistics Education* 27.2 (2019), pp. 110–119.

[Fos+11] Marc Fossi et al. “Symantec internet security threat report trends for 2010.” In: *Volume XVI* (2011).

[Fou17] Philippe et al. Fournier. “A survey of itemset mining.” In: *DMKD* 7.4 (2017), e1207. DOI: 10.1002/widm.1207. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1207. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1207.

[Fre19] Jonathan Freeman. *What is JSON? A better format for data exchange*. https://www.infoworld.com/article/3222851/what-is-json-a-better-format-for-data-exchange.html. Oct. 2019.

[Gér17] A. Géron. “Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.” In: O’Reilly Media, 2017. ISBN: 9781491962244. URL: https://books.google.dz/books?id=bRpYDgAAQBAJ.

[GP+14a] Ibrahim Ghafir, Vaclav Prenosil, et al. “Advanced persistent threat attack detection: an overview.” In: *Int J Adv Comput Netw Secur* 4.4 (2014), p. 5054.

[GP+14b] Ibrahim Ghafir, Vaclav Prenosil, et al. “Advanced persistent threat attack detection: an overview.” In: *Int J Adv Comput Netw Secur* 4.4 (2014), p. 5054.

[Hah19] Michael Hahsler. *Association rules measures: https://michael.hahsler.net/research/association_* 2019. URL: https://michael.hahsler.net/research/association%5C_rules/measures.html.

[HPS18] Xueyuan Han, Thomas Pasquier, and Margo Seltzer. “Provenance-based intrusion detection: opportunities and challenges.” In: *10th USENIX Workshop on the Theory and Practice of Provenance (TaPP 2018)*. 2018.

[HOF18] CHRIS HOFFMAN. *What Is a CSV File, and How Do I Open It?* https://www.howtogeek.com/348960/what-is-a-csv-file-and-how-do-i-open-it/. Apr. 2018.

[Hua12] David et al. Huang. “Rare Pattern Mining on Data Streams.” In: *Proceedings of the 14th International Conference on Data Warehousing and Knowledge Discovery*. DaWaK’12. Vienna, Austria: Springer-Verlag, 2012, pp. 303–314. ISBN: 978-3-642-32583-0. DOI: 10.1007/978-3-642-32584-7_25. URL: http://dx.doi.org/10.1007/978-3-642-32584-7_25.

[JK02] Kalervo Järvelin and Jaana Kekäläinen. “Cumulated gain-based evaluation of IR techniques.” In: *ACM Transactions on Information Systems (TOIS)* 20.4 (2002), pp. 422–446.

[JTJ20] Lianchao Jin, Fuxiao Tan, and Shengming Jiang. "Generative adversarial network technologies and applications in computer vision." In: *Computational Intelligence and Neuroscience* 2020 (2020).

[Kha22] Muhammad Asad Iqbal Khan. *Anomaly Detection with Isolation Forest and Kernel Density Estimation*. https://machinelearningmastery.com/anomaly-detection-with-isolation-forest-and-kernel-density-estimation/. Jan. 2022.

[KVV04] Christopher Kruegel, Fredrik Valeur, and Giovanni Vigna. "Intrusion detection and correlation: challenges and solutions." In: vol. 14. Springer Science & Business Media, 2004.

[Ksh10] Nir Kshetri. "The global cybercrime industry: economic, institutional and strategic perspectives." In: Springer Science & Business Media, 2010.

[Lal+18] Tarun Lalwani et al. "Implementation of a Chatbot System using AI and NLP." In: *International Journal of Innovative Research in Computer Science & Technology (IJIRCST) Volume-6, Issue-3* (2018).

[Més15] Eva Mészárosová. "Is python an appropriate programming language for teaching programming in secondary schools." In: vol. 4. 2. 2015, pp. 5–14.

[Mit97] Tom M. Mitchell. "Machine Learning." In: 1997, p. 1.

[Num22] the NumPy community. "NumPy User Guide Release 1.22.4." In: 2022, p. 7.

[Pas99] Nicolas et al. Pasquier. "Discovering Frequent Closed Itemsets for Association Rules." In: *Proceedings of the 7th International Conference on Database Theory*. ICDT '99. London, UK, UK: Springer-Verlag, 1999, pp. 398–416. ISBN: 3-540-65452-6. URL: http://dl.acm.org/citation.cfm?id=645503.656256.

[PED21] YAMINI PEDDIREDDI. *Dealing with Anomalies in the data*. https://www.analyticsvidhya.com/blog/2021/04/dealing-with-anomalies-in-the-data/. Apr. 2021.

[Poi19] Tutorials Point. *Google Colab - What is Google Colab*. https://www.tutorialspoint.com/google_colab/what_is_google_colab.htm. 2019.

[RDR12] Terry R Rakes, Jason K Deane, and Loren Paul Rees. "IT security planning under uncertainty for high-impact events." In: *Omega* 40.1 (2012), pp. 79–88.

[RFA21] Yousra Regaya, Fodil Fadli, and Abbes Amira. "Point-Denoise: Unsupervised outlier detection for 3D point clouds enhancement." In: *Multimedia Tools and Applications* 80 (July 2021), pp. 1–17. DOI: 10.1007/s11042-021-10924-x.

[Sam59] Arthur L Samuel. "Machine learning." In: *The Technology Review* 62.1 (1959), pp. 42–45.

[Sha21] Mohammed Yousef Shaheen. "Applications of Artificial Intelligence (AI) in healthcare: A review." In: *ScienceOpen Preprints* (2021).

[Tan11] Colin Tankard. "Advanced persistent threats and how to monitor and deter them." In: *Network security* 2011.8 (2011), pp. 16–19.

[tea22] Domino team. *Anaconda*. https://www.dominodatalab.com/data-science-dictionary/anaconda/. JUIN 2022.

[TB08] Ciza Thomas and N Balakrishnan. "Improvement in minority attack detection with skewness in network traffic." In: 6973 (2008), 69730N.

[Ull+20] Zaib Ullah et al. "Applications of Artificial Intelligence and Machine learning in smart cities." In: *Computer Communications* 154 (2020), pp. 313–323. ISSN: 0140-3664. DOI: https://doi.org/10.1016/j.comcom.2020.02.069. URL: https://www.sciencedirect.com/science/article/pii/S0140366419320821.

[Zak00] Mohammed J. Zaki. "Scalable Algorithms for Association Mining." In: *IEEE TKDE* 12.3 (May 2000), pp. 372–390. ISSN: 1041-4347. DOI: 10.1109/69.846291. URL: http://dx.doi.org/10.1109/69.846291.

[Zak02] Mohammed Javeed et al. Zaki. "CHARM: An Efficient Algorithm for Closed Itemset Mining." In: *SDM*. SIAM, 2002, pp. 457–473. ISBN: 978-1-61197-272-6. URL: http://dblp.uni-trier.de/db/conf/sdm/sdm2002.html#ZakiH02.