University of Thessaly
Greece, Spring 2021

Machine Learning techniques for a successful intrusion detection system: a survey

Τεχνικές μηχανικής μάθησης για επιτυχημένα συστήματα ανίχνευσης απειλών

Sotirios Panagiotis Chytas

Supervisor: George Stamoulis

Master's Thesis

Department of Computer Sciences

University of Thesssaly

Volos, Greece

This Thesis was written as part of the requirements for the Master's degree of Computer Sciences at University of Thessaly.

Declaration of Authorship

I, Sotirios Panagiotis Chytas, declare that this thesis titled, "Machine Learning techniques for a successful intrusion detection system: a survey" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all the main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

University of Thessaly

Greece, January 2021

Sotirios Panagiotis Chytas

Περίληψη

Ολοένα και περισσότερα προβλήματα στον χώρο της τεχνολογίας (και όχι μόνο) επιλύονται με τεχνικές **Μηχανικής Μάθησης**. Ο τομέας αυτός, που είναι το πιο διάσημο υποσύνολο της Τεχνητής Νοημοσύνης, αποτελείται από μια πλειάδα διαφορετικών αλγορίθμων και τεχνικών, που συνεχώς αυξάνονται σε αριθμό.

Αυτήν η εργασία προσπαθεί να προσφέρει έναν πλήρη οδηγό χρήσης αυτών των τεχνικών, με επεξήγηση των διαθέσιμων επιλογών, με σκοπό να χρησιμοποιηθεί από τους ερεύνητες και προγραμματιστές στον τομέα των δικτύων που επιθυμούν να ενσωματώσουν στα συστήματα τους ένα σύστημα ανίχνευσης απειλών που βασίζεται σε τεχνικές μηχανικής μάθησης.

Το μεγαλύτερο μέρος της μεταπτυχιακής αυτής διατριβής έχει παρουσιαστεί και σε συνέδριο και απέσπασε το βραβείο της καλύτερης μαθητικής εργασίας [1].

Abstract

Machine learning offers solutions to a great number of technological (and not only) problems; a number that constantly increases. Machine learning, the most famous AI subset, consists of a growing number of different algorithms and techniques.

This work aims to offer a complete guide of all these techniques, along with insights on the different design choices. Our goal is to provide with guidance the network researchers and programmers who want to incorporate a machine learning based intrusion detection system, in their platforms.

The biggest part of this work has already been submitted to a conference and received the best student paper award [1].

Acknowledgments

Firstly, I would like to thank my thesis supervisor associate professor George Stamoulis for his guidance throughout my journey of thesis writing. With his help I managed to present a complete review of a constantly growing field, and receive a precious award for our effort.

Moreover, I should thank all these people who believed in me and helped me (even without realizing) and became a part of me. I am particularly thankful to my parents for their unconditional love and support. Thank you for bearing with me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years cyber attacks, especially those targeting Critical National Infrastructures that provide essential information or services are becoming more sophisticated and difficult to detect. The protection of their network and information systems becomes a significant issue [7]. The attacks on such systems vary from reconnaissance attacks, to attempts of penetrating to the internal network and installation and execution of malicious code that can be used in order to steal sensitive data or even change the behavior of specific physical equipment with devastating consequences. A category of attacks, entitles Advanced Persistent Threats (APTs) are stealthy attacks, with the ability to stay undetected, concealing themselves within targets network, and interacting just enough to achieve the defined objectives. For example, APT actors may use zero-day exploits to avoid signature-based detection, and encryption to obfuscate network traffic.

In order to tackle this growing trend academia and industry are joining forces in an attempt to develop novel systems and mechanisms that can defend their systems. Along with other preventive and reactive security tools and solutions that are proposed, such as movel access control and authentication mechanisms, intrusion detection systems (IDS) are deployed as a second line of defense. IDSs can distinguish between normal and malicious actions [8] using either specific rules or patterns of normal behavior of the system.

An Intrusion Detection System (IDS) is a device or software application that monitors a network or system for malicious activity or policy violations. Intrusion detection and prevention systems are primarily focused on identifying possible incidents in systems and

networks, logging information, reporting attempts, and learning. IDS have become a necessary addition to the security infrastructure of nearly every organization [9]. Based on the current practical situations, machine learning (ML) technologies have been employed to improve the efficiency of intrusion detection systems, which is one of the most commonly used security infrastructures to protect networks from attacks.

Recently many novel algorithms have been proposed. In [10] authors proposed a hierarchical intrusion detection system based on the combination of three different classifiers, the REP Tree, the JRip algorithm and Forest PA, that can be used for IoT. In another work, authors in [11] proposed a novel IDS that incorporates physical characteristics in order to detect spoofing attacks in a networks of connected vehicles. In a recent survey work [8] the importance of deep learning techniques in detecting novel attacks was revealed.

In this work, we focus on network-level threats and we present the most prominent machine learning ideas that can solve that problem. We use the CICIDS2017 dataset, a huge dataset with more than 2.8 million rows of benign and malignant cases.

Usually IDS are based on [12; 13]:

1. Signature-based detection (Compare against saved signatures. Does not generalize to new threats.)
2. Anomaly-based detection (Detect any deviations from normal behaviours based on data. Does generalize.)
3. Stateful protocol analysis detection (Compare against pre-determined behaviours and patterns. Also does not generalize to new threats.)

Machine learning techniques belong to the second category (Anomaly Detection), and this is the category we are going to analyze.

Although there are many works that present various machine/deep learning algorithms that can be used for an IDS [14; 15; 16; 17; 18; 19], it seems to be a shortage of papers that analyze possible ways to use these algorithms to build an IDS based on various cases of needs and resources. Our goal is to provide all the well-known techniques that can be used for an IDS. This work, in combination with papers that offer a survey on specific

machine learning algorithms [8], is a valuable document for any scholar who wants to build or customize an IDS based on machine/deep learning techniques.

1.1 Thesis Structure

The remainder of the thesis is organized as follows:

- In chapter 2, we provide a brief introduction to the main Machine Learning algorithms that are used today. We mention the basic ideas of each method as well as the underlying maths.
- In chapter 3, we explain how the algorithms presented above can be used in a real-life system, as well as the most common tuning practices that are particularly useful in an Intrution Detection System.
- In chapter 4, we present the dataset we use for the testing in all the experiments.
- In chapter 5, we present the results of our system, alongside with graphs that provide more insight about the design choices.
- Finally, in chapter 6, we provide a conclusion to this thesis.

Chapter 2

Machine Learning Algorithms

Machine learning is a subset of the vast field called *Artificial Intelligence* (AI). In computer science, artificial intelligence, sometimes called machine intelligence, is intelligence demonstrated by machines, in contrast to the natural intelligence displayed by humans and animals. Colloquially, the term "artificial intelligence" is used to describe machines that mimic "cognitive" functions that humans associate with other human minds, such as "learning" and "problem-solving" [20].

Machine Learning or Statistical Learning, uses statistical methods in order to "learn" from data and generalize to new, unseen examples.

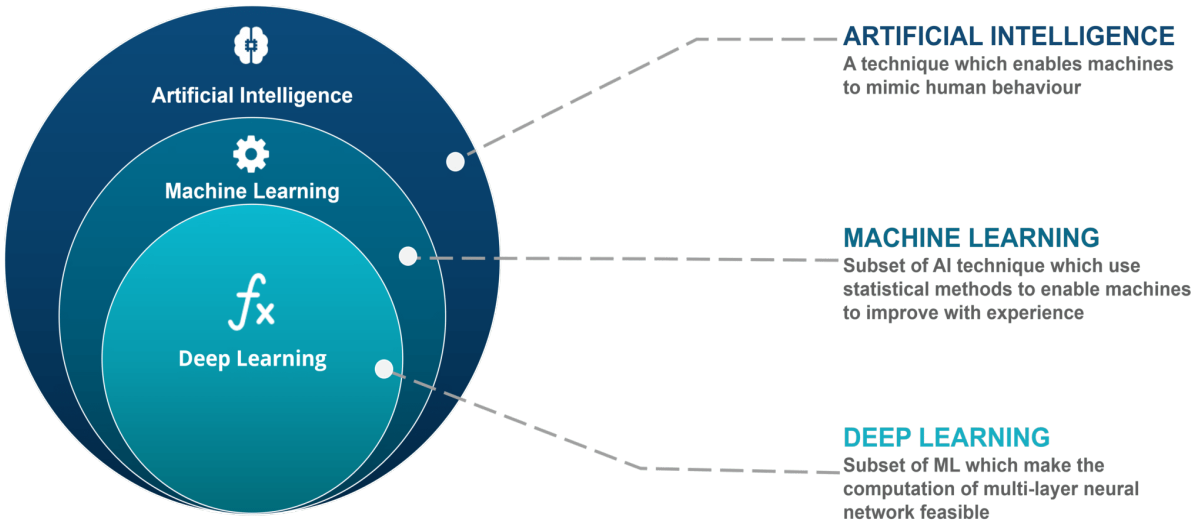

Figure 2.1: AI, Machine Learning, and Deep Learning (Figure from [21])

2.1 Overview

Machine learning is a data-driven approach. In contrast to old-school ai methods that involved rules and a world model [22], machine learning tries to learn only from labeled data with no other information.

Mathematically, given pairs of data X, y with X being a matrix whose i_{th} row depicts the data of the i_{th} example (e.g. size of incoming packet, time delay, ...) and y being a vector whose i_{th} element is the label of the corresponding row in the X matrix (e.g. benign packet or malignant), we try to find a function f that approximates the data; i.e. $f(X_i) \simeq y_i$.

The two most common machine learning problems are (1) classification and (2) regression problems. In regression problems $y \in \mathbb{R}$ while in classification, y takes only some specific values (e.g. True/False, Category 1/Category 2/.../Category n). This work focuses on classification algorithms, as we deal with Intrusion Detection Systems.

2.2 Common algorithms

2.2.1 Logistic Regression

The most common, and simplest classification algorithm is the Logistic Regression [23]. Assuming $X_i \in \mathbb{R}^n$, then we try to find a hyperplane in the n-dimensional space that separates the two classes. In the two dimensional space, we can visualize the result as a separating line (Fig. 2.2).

Mathematically, we can write this as:

$$f(X_i) = \sigma(w^T X_i + b) \tag{2.1}$$

with σ being the sigmoid function

$$\sigma = \frac{1}{1 + e^{-x}} \tag{2.2}$$

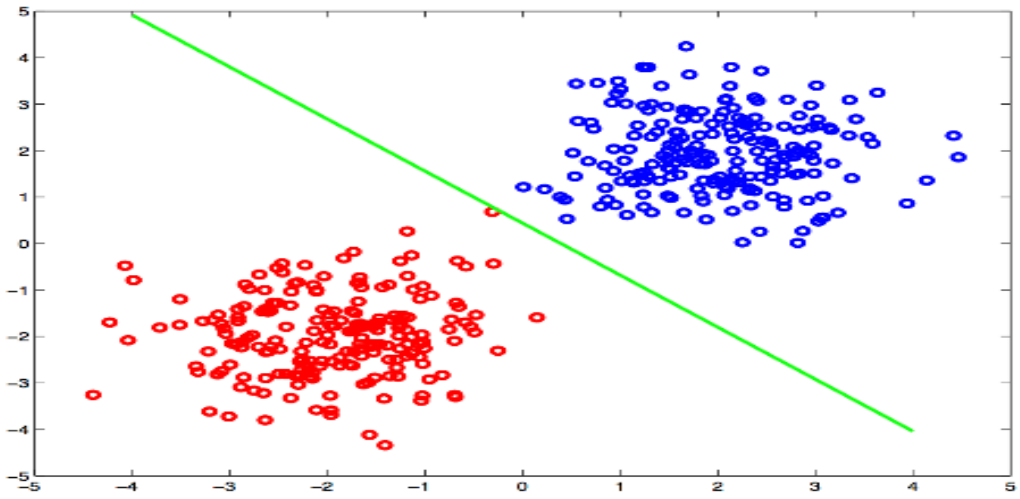

Figure 2.2: Separation of the two classes using Logistic Regression (Figure from [2])

2.2.2 Support Vector Machine

Support vector machine (SVM) [24] can be considered an extension/improvement of the simple logistic regression, although the background mathematics are far more complicated. Specifically, the underlying idea of SVM is to find the *best* separating line between the two classes (Fig 2.3).

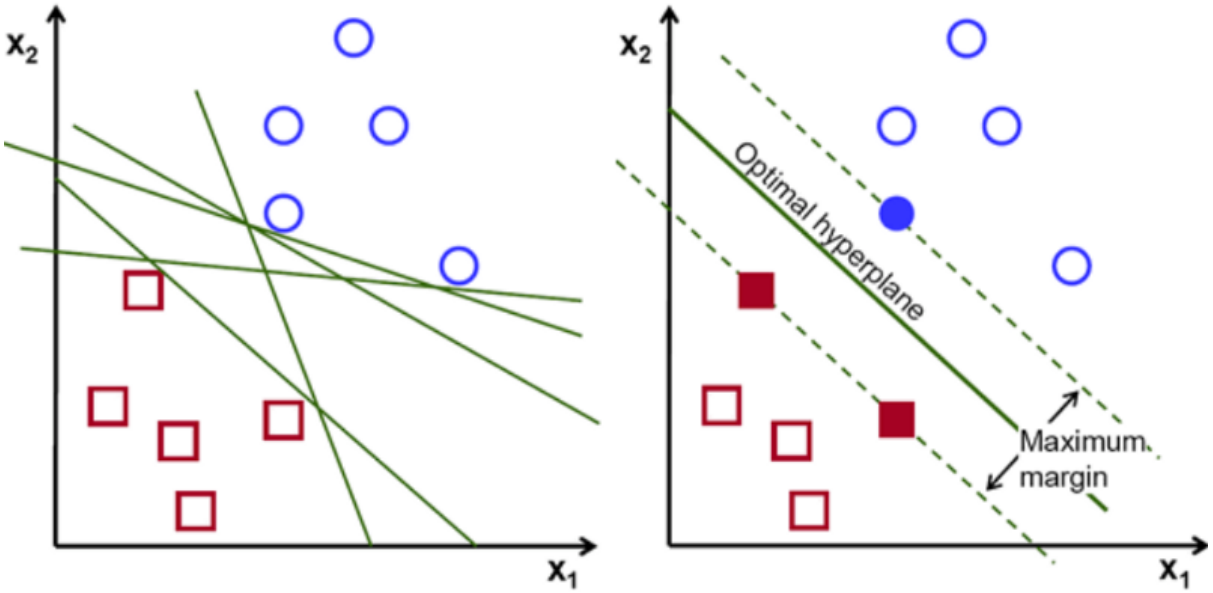

Figure 2.3: Logistic Regression vs Support Vector Machine. Logistic Regression may end up to any of the separating lines depicted in the left side, while SVM will try to find the one line that maximizes the space in between. (Figure from [3])

2.2.3 Decision Tree

One of the most famous machine learning algorithms is the Decision Tree [25]. Decision Trees can achieve high performance while being easy to understand from a human. This method slices recursively the n-dimensional space of the data, into two sub-regions at each step. The final space looks like on the left of Fig. 2.5, but it is easier to understand by looking on the right side of Figure.

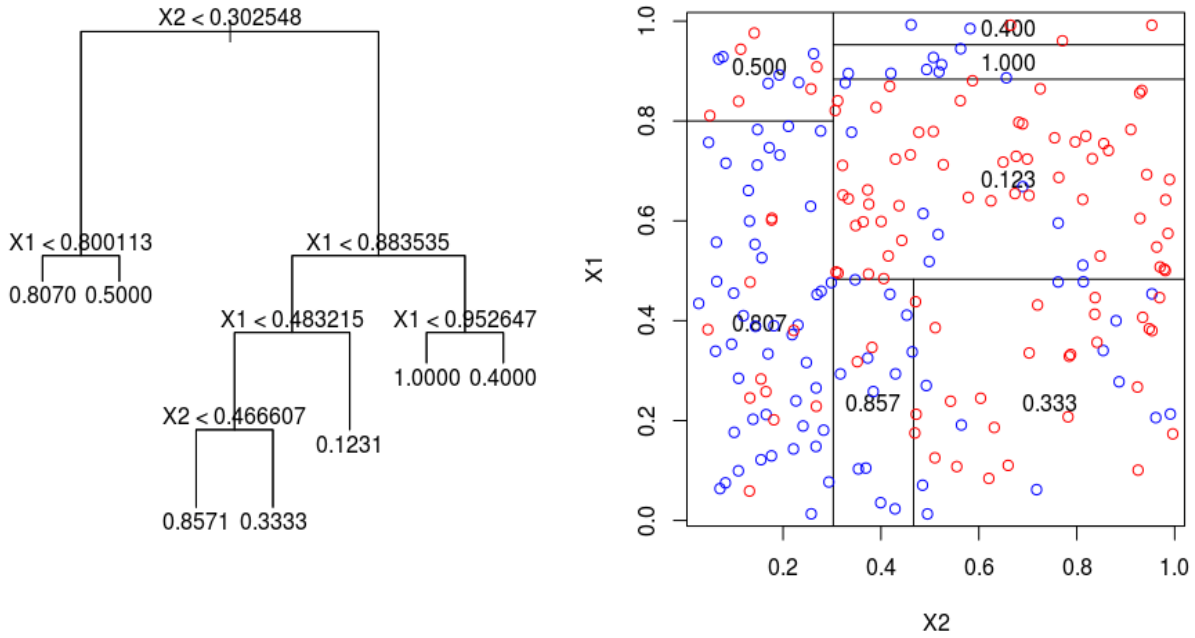

Figure 2.4: Decision Tree. Its structure and the corresponding decision space (Figure from [4])

A famous extension of the Decision Trees is the Random Forest [26] which combines many such trees and achieves superior performance (Sec. 2.3).

2.2.4 Neural Networks

The most famous algorithm of the last decade, Neural Networks [27] have led to amazing (and sometimes even superhuman) performance in a variety of machine learning tasks. Neural Networks consist of neurons, which closely resemble the logistic regression algorithm. Specifically, the output of each neuron is given by

$$o = f(w^T X_i + b) \tag{2.3}$$

with f being any function we want (of course, some options are better than others though).

The main advantage of the Neural Networks, is the ability to combine many such simple neurons, and achieving amazing performance. A typical feedforward network is depicted below.

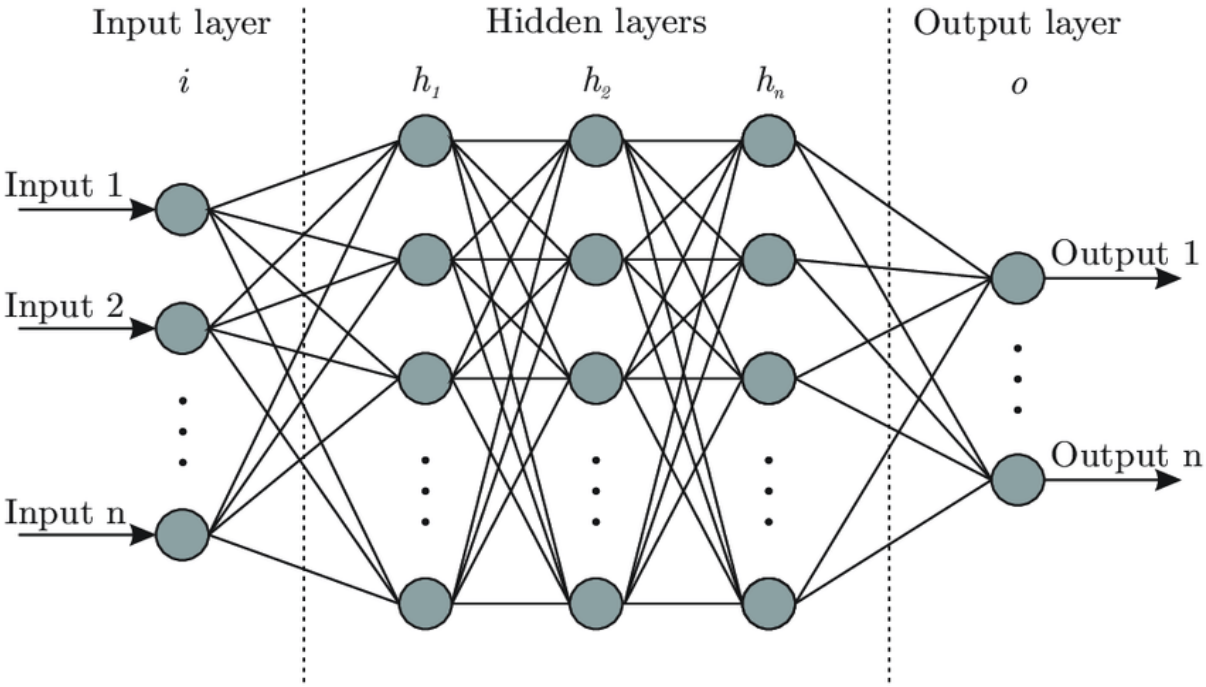

Figure 2.5: A typical feedforward neural network. Neurons are organized in layers. (Figure from [5])

2.3 Ensemble methods

An extension to all these algorithms the presented above is the so called Ensemble Learning [28]. Ensemble Learning is a set of techniques (e.g. boosting, bagging) that combine efficiently many different instances of the above algorithms (mainly Decision Trees), in order to achieve better performance. The two most famous algorithms are Random Forests [26] and Gradient Boosting [29].

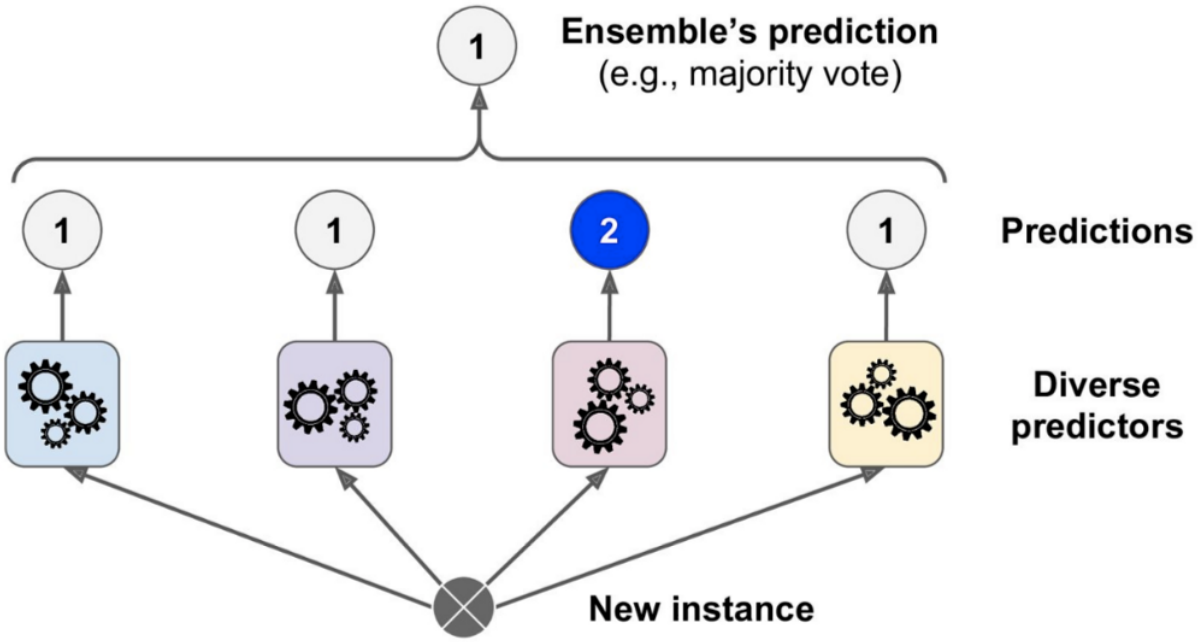

Figure 2.6: A high-level overview of the ensemble learning process. (Figure from [6])

Chapter 3

Machine Learning in practice

3.1 Binary Classification

As we already mentioned, the focus of this work is on classification (i.e. classify an example among some specific, known classes). Usually, in an IDS system, our main focus is classifying an activity between Benign or Malignant. This particular type of classification is called two-class or binary classification. An extension to these models is the so-called multi-class classification. In this case, we deal with more than two classes (e.g. Benign, DDOS attack, PortScan, etc.).

Our main focus is to create an accurate binary classifier (Benign vs Malignant). A multi-class classifier that distinguishes between the different types of attacks can be used after that, but the IDS's main component should absolutely be the detection of any Malignant cases (no matter the particular type). Below we will present all these the design choices that occur in a binary classifier and can be very helpful in an IDS.

3.1.1 Vanilla Binary classification

The simplest case is to use a model presented before without much further tuning. In this case, we assume that both classes are equally important and that the dataset is not highly imbalanced. As it is obvious, in an IDS system none of these assumptions usually hold. The Malignant cases are scarce (imbalanced) and we care much more for detecting these cases. However, such simpler models should be used/tested first as they may be

more that satisfactory for our needs.

3.1.2 Weighted Binary classification

As we mentioned before, a typical binary classifier assumes that both classes are equally important. This means that, in a balanced dataset, the False Positives will be approximately equal to the False Negatives, or, to put it in other terms, the model is not biased towards one class. In the case of a highly imbalanced dataset, the model will be biased towards the dominant class.
Obviously, such behaviour is very different from an ideal IDS. An IDS should be more "careful" (i.e. biased) with the Malignant cases, even if this means that some Benign cases will be classified as Malignant. The reason for that is that an attack can be very destructive in both financial and privacy terms [30].

Thankfully, many algorithms provide a weighted classification option. Visually, we can think of weighted binary classification as a stretch of the "valuable" class' region. For example, in Fig. 3.1, if we value more the red class, we stretch the red region and make it bigger. The bigger the weight, the bigger (theoretically) that stretch is. We can clearly see that there is a trade-off between correctly classified red points, and misclassified blue points.

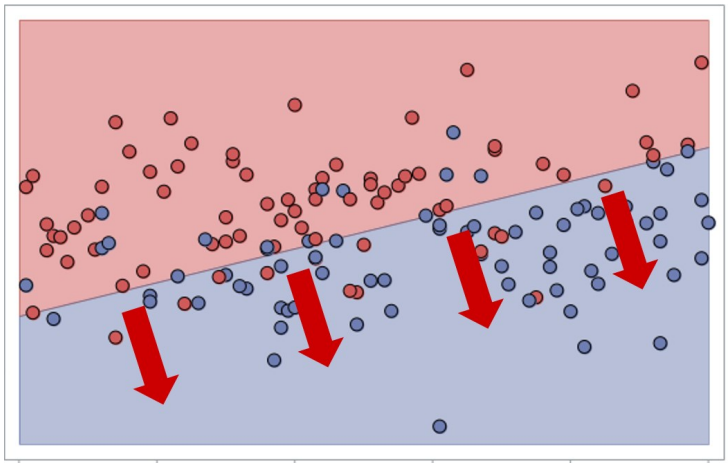

Figure 3.1: The effect of the weighted binary classification to the decision boundary of the unweighted bimary classification. The arrows represent the move of the boundary in favor of the red class. The bigger the red class' weight, the bigger the arrows.

3.1.3 Threshold tuning

All the methods output a number between 0 and 1 that resembles to the probability p of the one class (the other class's probability is 1-p). So, to further improve the results of the algorithms, we can tune the probability threshold. By default, when a classifier outputs a probability of benign higher of 0.5 then we predict that this case is a benign one and malignant otherwise. However, we can alter this threshold. For example, we may set it to 0.7 if we value more the malignant cases, or to a value lower than 0.5 if we care more for the benign cases.

However, in order to properly tune the threshold without overfitting to the testing dataset, we need a new, unseen dataset. The training pipeline is depicted in Fig. 3.2.

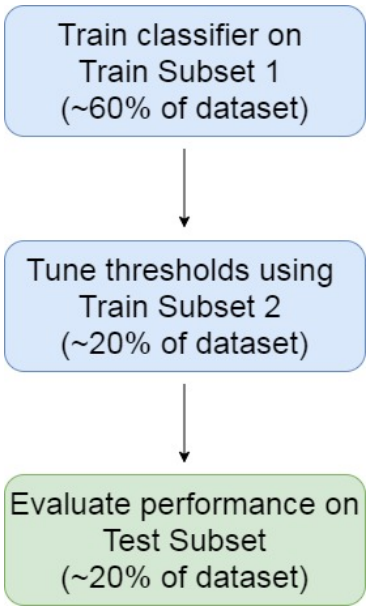

Figure 3.2: Threshold tuning pipeline

3.2 One-Class Classification

In many real-life scenarios, we may have access to only one of the two classes of interest. This is very common in IDS. Creating attack scenarios requires a lot of resources, while there is an infinite amount of Benign cases.

In this case, we can use the so called one-class classifiers. One-class classification is synonymous to anomaly detection. This methods try to learn the form of the given class so that they are able to detect examples that are very different from this form. However,

this is a much harder task than binary classification, because the model's "world" is just this given class. In simple terms, the most common technique is to understand the true values of each feature as well as for combinations of these features, so that they will be able to detect any big deviations from these ranges, and mark these cases as outliers (malignant cases). Consequently, such algorithms perform much worse than the binary classifiers presented above.

3.2.1 AutoEncoders

Many recent works [31; 32] have explored the idea of using an autoencoder to detect any anomalies. An autoencoder architecture trained on normal data (benign) is very likely to not be able to properly reconstruct a "weird" example (malignant). The big advantage of this idea is the use of only benign cases (which are abundant in contrast to malignant ones) during the training phase.
The two basic principles of the autoencoders are 3.3:

1. Dimensionality reduction
2. Reconstruction

By reducing the samples' dimension we, hopefully, keep only the important, meaningful parts. Then, we can reconstruct each using this, reduced representation. In the case of an outlier/anomaly the reconstruction error will then be much higher.

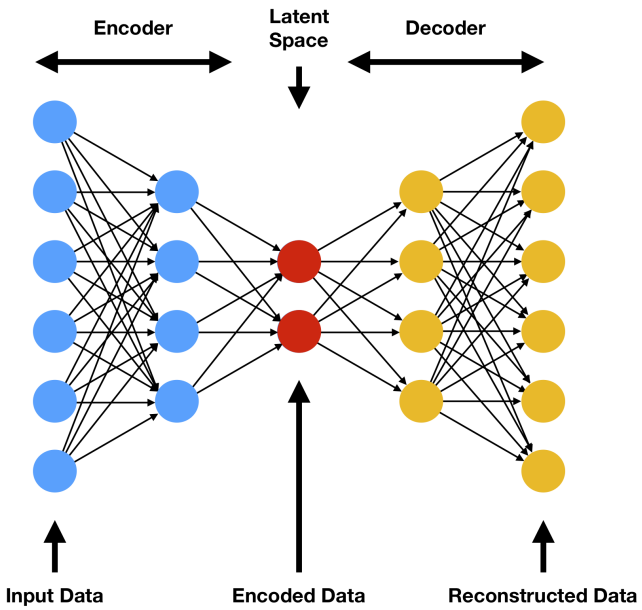

Figure 3.3: The typical autoencoder's architecture. Both dimensionality reduction (latent space) and reconstruction (reconstructed data) are obvious.

Chapter 4

The CIC-IDS2017 dataset

4.1 Overview

Although the techniques presented in this work are applicable to any dataset, we evaluate all methods to a specific dataset, the CICIDS2017 [33]. It is one of the largest intrusion detection datasets, with more than 2.8 millions rows, splitted into 8 smaller datasets. The complete data generation process can be found in [34].

4.2 Labels

The dataset contains data for 14 different attacks (e.g. DDOS, XSS, SQL-injection, etc) and normal cases (Benign). The distribution of the labels is provided in Fig. 4.1. However, a typical IDS system usually cares for classifying between benign and malignant cases, and not distinguish between the different attack types. For this reason, we change the names of the labels to BENIGN (0) and MALIGNANT (1) as we also care about building an Intrusion Detection System an not an Intrusion Classifier. About 80% of the dataset consists of Benign cases and the rest 20% is various Malignant cases.

4.3 Features

Each of these 2.8 millions rows consists of 78 features. Some of these features are:

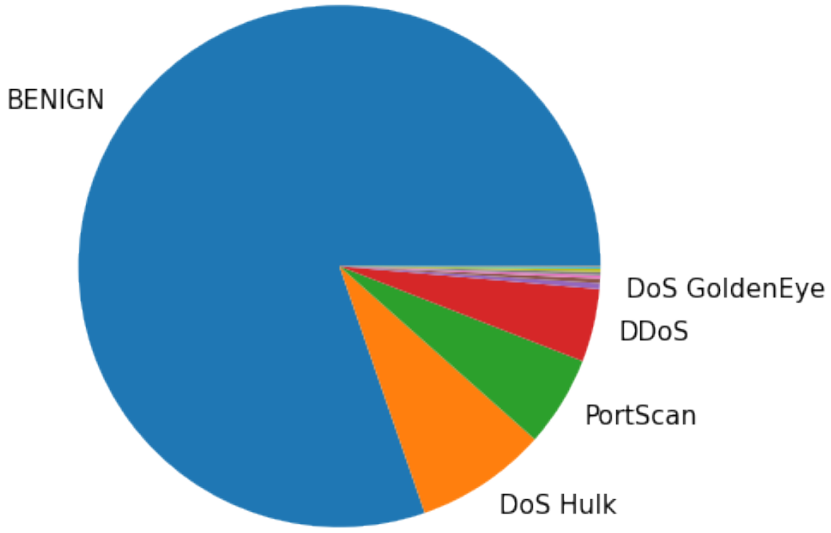

Figure 4.1: Labels distribution on CIC-IDS2017. Clearly, the dataset is highly unbalanced.

- Destination Port
- Flow Duration
- Total Fwd Packets
- Total Backward Packets
- Total Length of Fwd Packets
- Total Length of Bwd Packets

However, in all of the models we work with only 52 of these. The reason behind this is the following two preprocessing steps:

1. Remove any columns (features) whose most frequent value is encountered in more than 99.9% of the rows.
2. Remove one of the two columns of each pair if their absolute correlation is more than 0.99.

Obviously, someone can work without applying these steps. But, there will be no increase in the performance while the training time will increase (by a lot in some cases). Also, more features mean a higher danger of overfitting.

Chapter 5

Results

5.1 Baseline - Null model

Before any modeling attempts, we need to define a naive model as well as the corresponding accuracy. In our case, a naive model is a model that can not detect any intrusions. Perhaps surprisingly, such a model achieves 80% accuracy, as we deal with an imbalanced dataset that consists mostly (80%) of Benign cases.

5.2 Evaluation

In all subsequent models, we used 80% of the dataset (2.25 million rows) for training, and the rest 20% for evaluation.

5.3 Binary classification

A binary classifier is the simplest model we can have; a model that assumes both classes are equally important. The results (accuracy and f-score) for all models are depicted in Table 5.1. It is obvious that tree-based algorithms dominate the leaderboard.

Table 5.1: Accuracy and F-score results for the most well-known binary classification algorithms. (neural network's capacity against performance is depicted in more detail, in Subsection 5.3.1).

Algorithm	Accuracy	F-score
Logistic Regression	93.5%	83.4%
Decision Tree	99.87%	99.7%
Random Forest	**99.89%**	**99.74%**
Gradient Boosting	99.67%	99.2%
Neural Network	99.5%	98.7%

A more in depth understand of all models' performance can be obtained by directly examining the False Positives (FP), and False Negatives (FN) instead of a another metric. For that reason, in Table 5.2 we depict all of them, for all trained models.

Table 5.2: False positives & false negatives. False positives measure refers to benign cases that were wrongly classified as malignant and false negatives to incorrectly classified true malignant cases.

Algorithm	False Pos	False Neg
Logistic Regression	16696	19827
Decision Tree	345	350
Random Forest	**297**	**280**
Gradient Boosting	913	942
Neural Network	1824	1100

5.3.1 Neural Networks

The change on Neural Network's accuracy as we increase the network's hidden layers is depicted in Fig. 5.1. We observe that the improvement is negligible. However, the training time explodes. In the last case (4 hidden layers) the training time is approximately the same as in the previous case because the network overfits quickly and we used the early stopping technique to prevent that. (We used 100 neurons in each hidden layer. Different sizes lead to similar results).

5.4 Weighted binary classification

As we already mention, weighted classification can be used when we care more for one class, even if this hurts the overall model's performance (in terms of accuracy etc.). IDS

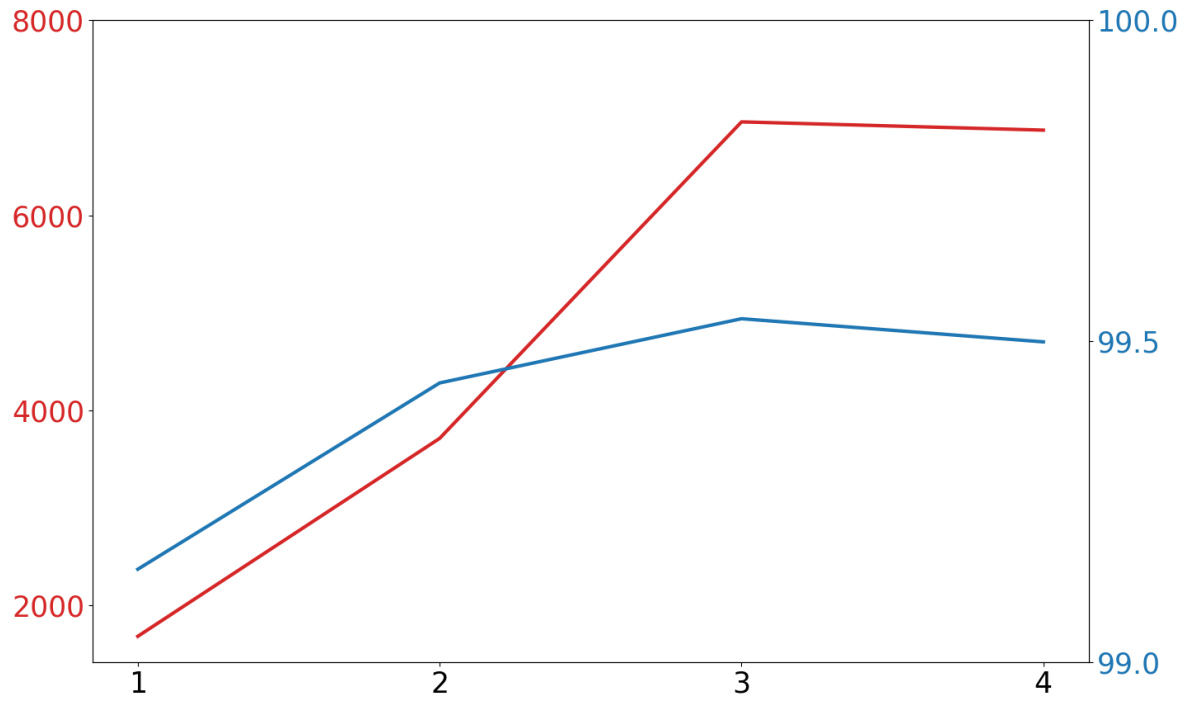

Figure 5.1: The change on Neural Network's accuracy (right y-axis) as we increase the network's hidden layers (x-axis). On the left y-axis we depict the training time in seconds.

is the most prominent example of such a case. We care much more for correctly detecting any intrusions even if this would lead to labeling a few benign cases as malignant.

However, not all algorithms provides us with a weighted option. Thankfully, tree-based methods (the ones that perform the best on the simple binary classification) have such functionality. The results in terms of FP and FN are depicted in Table 5.3.

Table 5.3: False positives and false negatives for the weighted cases. Inside the parenthesis we present the results of the previous section i.e. same weights. The decrease of the false negatives is obvious as well as the increase of the false positives.

Algorithm	False Pos	False Neg
Logistic Regression	27355 (16696)	8785 (19827)
Decision Tree	355 (345)	**242** (350)
Random Forest	**345** (297)	259 (280)

The results were obtained by setting the weight of the Malignant cases to 2 and that of Benign to 1. A more in depth exploration of the weight effect is given in Fig. 5.2.

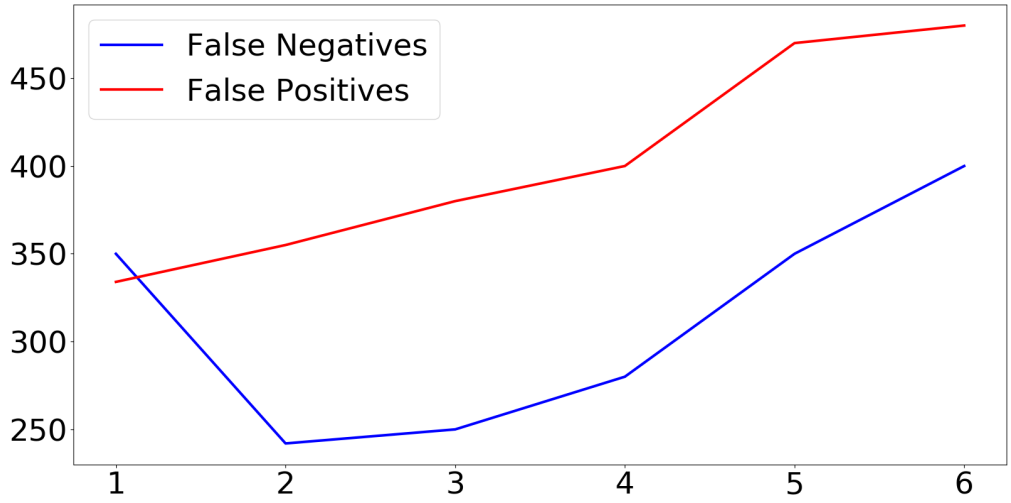

Figure 5.2: The change on FN and FP (y-axis) based on the weight (x-axis) we apply to the malignant class (benign has always weight=1). We can see that too big values lead to worse results for both classes. We used Weighted Decision Trees for the graph.

5.5 Feature importance

Feature importance can play a vital role in updating our dataset and creating a lighter, generalizable model. The 5 most prominent features are depicted in Fig. 5.3 for each of the three models.

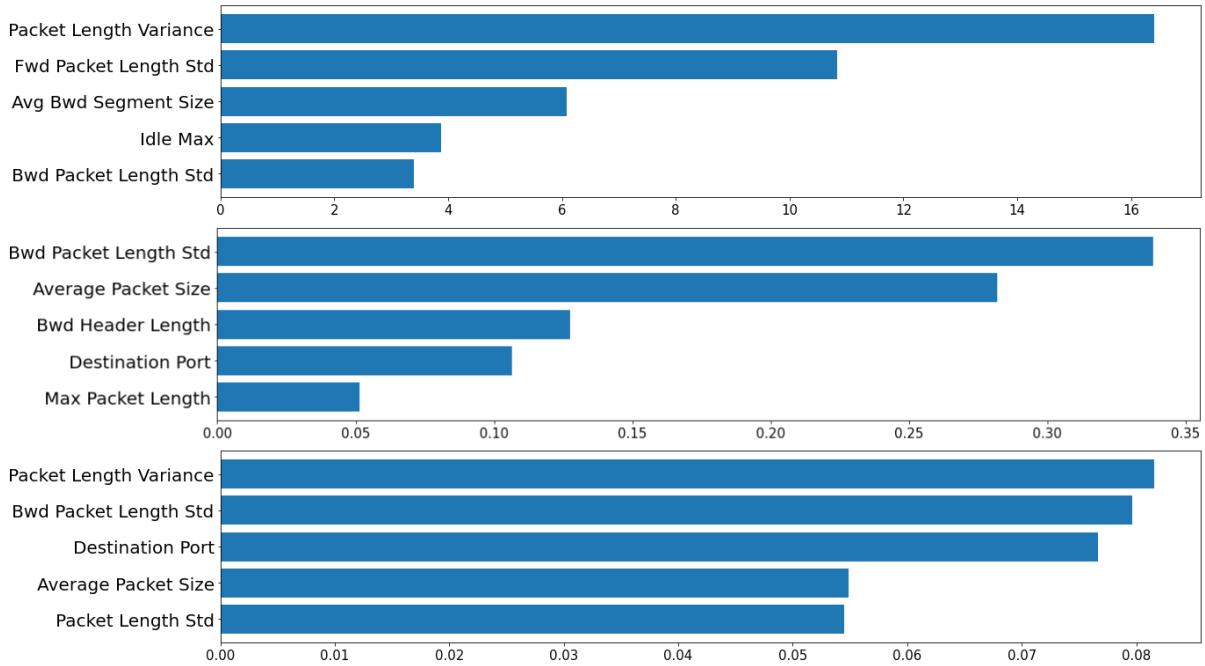

Figure 5.3: From top to bottom. (1) Top-5 absolute coefficients of the Logistic Regression method. (2) Top-5 Weighted Decision Tree importances. (3) Top-5 Weighted Random Forest importances.

Chapter 6

Conclusion

In this thesis, we presented a complete guide for anyone who wants to build an Intrusion Detection System with data-driven Machine Learning techniques. Moreover, we carefully analyzed all the possible design choices and explaing the reason behind each one of these, so that anyone is able to design a custom IDS that fits his/hers needs.
Everything was tested against the CIC-IDS2017 dataset, the largest and most famous IDS dataset, and the results were more than satisfactory. The whole design pipeline, for future reference, is depicted in Fig. 6.1

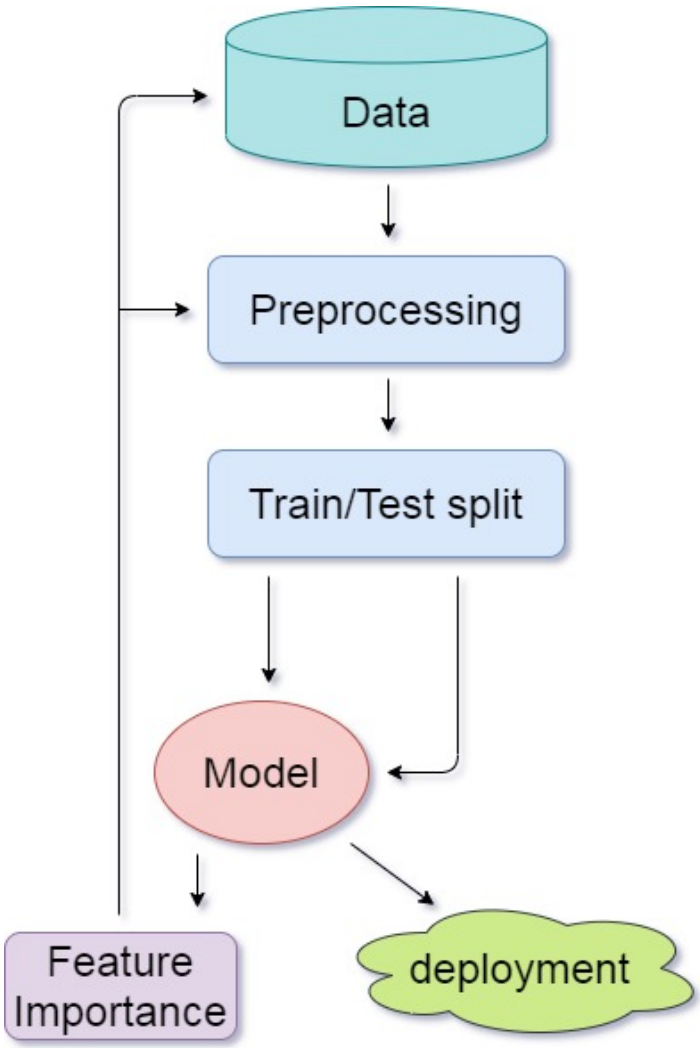

Figure 6.1: The whole training pipeline. After preprocessing our data, we train the algorithms, and use feature importance to improve the results by altering the dataset

Bibliography

[1] S. Chytas, L. Maglaras, A. Derhab, and G. Stamoulis, "Assessment of machine learning techniques for building an efficient ids," 07 2020.

[2] "Notes Logistic Regression - Standford ML Andrew Ng," https://joparga3.github.io/standford_logistic_regression/.

[3] "Support Vector Machine vs Logistic Regression," https://gdcoder.com/support-vector-machine-vs-logistic-regression/.

[4] "Decision Trees in R," https://www.datacamp.com/community/tutorials/decision-trees-R.

[5] "Artificial Neural Networks Demystified," https://medium.com/@gabriel.mayers/artificial-neural-networks-demystified-d7cbfb6c6916.

[6] "Ensemble Learning: 5 Main Approaches," https://www.kdnuggets.com/2019/01/ensemble-learning-5-main-approaches.html.

[7] L. A. Maglaras, K.-H. Kim, H. Janicke, M. A. Ferrag, S. Rallis, P. Fragkou, A. Maglaras, and T. J. Cruz, "Cyber security of critical infrastructures," *Ict Express*, vol. 4, no. 1, pp. 42–45, 2018.

[8] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, p. 102419, 2020.

[9] M. Martellini and A. Malizia, *Cyber and Chemical, Biological, Radiological, Nuclear, Explosives Challenges: Threats and Counter Efforts*, 2017.

[10] M. A. Ferrag, L. Maglaras, A. Ahmim, M. Derdour, and H. Janicke, "Rdtids: Rules and decision tree-based intrusion detection system for internet-of-things networks," *Future Internet*, vol. 12, no. 3, p. 44, 2020.

[11] D. Kosmanos, A. Pappas, L. Maglaras, S. Moschoyiannis, F. J. Aparicio-Navarro, A. Argyriou, and H. Janicke, "A novel intrusion detection system against spoofing attacks in connected electric vehicles," *Array*, vol. 5, p. 100013, 2020.

[12] M. Whitman and H. Mattord, *Principles of Information Security*. Thomson Course Technology, 2009. [Online]. Available: https://books.google.gr/books?id=gPonBssSm0kC

[13] E. Kirda, S. Jha, and D. Balzarotti, "Recent advances in intrusion detection," *Proc. 12th International Symposium, RAID 2009, Saint-Malo, France.*, 9 2009.

[14] S. S. S. Sindhu, S. Geetha, and A. Kannan, "Decision tree based light weight intrusion detection using a wrapper approach," *Expert Systems with applications*, vol. 39, no. 1, pp. 129–141, 2012.

[15] S. A. Mulay, P. Devale, and G. Garje, "Intrusion detection system using support vector machine and decision tree," *International Journal of Computer Applications*, vol. 3, no. 3, pp. 40–43, 2010.

[16] P. B. Hasan M., Nasser M. and A. S., "Support vector machine and random forest modeling for intrusion detection system (ids)," *Journal of Intelligent Learning Systems and Applications*, vol. 6, no. 6, pp. 45–52, 2014.

[17] Y. Kim, M. S. Johnson, S. H. Knox, T. A. Black, H. J. Dalmagro, M. Kang, J. Kim, and D. Baldocchi, "Gap-filling approaches for eddy covariance methane fluxes: A comparison of three machine learning algorithms and a traditional method with principal component analysis," *Global Change Biology*, vol. 26, no. 3, pp. 1499–1518, 2020.

[18] S. A. R. Shah and B. Issac, "Performance comparison of intrusion detection systems and application of machine learning to snort system," *Future Generation Computer Systems*, vol. 80, pp. 157–170, 2018.

[19] S. K. Biswas, "Intrusion detection using machine learning: A comparison study," *International Journal of Pure and Applied Mathematics*, vol. 118, no. 19, pp. 101–114, 2018.

[20] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2009.

[21] https://www.edureka.co/blog/ai-vs-machine-learning-vs-deep-learning/.

[22] R. Davis and J. King, "The origin of rule-based systems in ai," 2005.

[23] J. S. Cramer, "The origins of logistic regression," 2002.

[24] C. Cortes and V. Vapnik, "Support-vector networks," in *Machine Learning*, 1995, pp. 273–297.

[25] X. Wu, V. Kumar, R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. Mclachlan, S. K. A. Ng, B. Liu, P. Yu, Z.-H. Zhou, M. Steinbach, D. Hand, and D. Steinberg, "Top 10 algorithms in data mining," *Knowledge and Information Systems*, vol. 14, 12 2007.

[26] Tin Kam Ho, "Random decision forests," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, 1995, pp. 278–282 vol.1.

[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[28] D. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *Journal of Artificial Intelligence Research*, vol. 11, p. 169–198, Aug 1999. [Online]. Available: http://dx.doi.org/10.1613/jair.614

[29] J. H. Friedman, "Greedy function approximation: A gradient boosting machine." *The Annals of Statistics*, vol. 29, no. 5, pp. 1189 – 1232, 2001. [Online]. Available: https://doi.org/10.1214/aos/1013203451

[30] "The Price of Security: How Much Does a Cybersecurity Attack Actually Cost?" https://dynasis.com/2019/03/price-security-how-much-cybersecurity-attack-actually-cost/.

[31] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *Special Lecture on IE*, vol. 2, no. 1, 2015.

[32] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, 2014, pp. 4–11.

[33] I. Sharafaldin, A. H. Lashkari, and A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *ICISSP*, 2018.

[34] "Intrusion Detection Evaluation Dataset (CIC-IDS2017)," https://www.unb.ca/cic/datasets/ids-2017.html.