

Master's thesis

# Game Theoretic Optimization of Detecting Malicious Behavior

*Raman Samusevich*



May 2016

Advisor: Mgr. Viliam Lisý, MSc., Ph.D.  
External co-advisor: Ing. Tomáš Pevný, Ph.D.

Czech Technical University in Prague  
Faculty of Electrical Engineering  
Department of Computer Science

Czech Technical University in Prague  
Faculty of Electrical Engineering

Department of Computer Science and Engineering

## DIPLOMA THESIS ASSIGNMENT

Student: **Bc. Raman Samusevich**

Study programme: Open Informatics  
Specialisation: Artificial Intelligence

Title of Diploma Thesis: **Game Theoretic Optimization of Detecting Malicious Behavior**

### Guidelines:

In domains of spam filtering, network intrusion detection, or fraud detection, machine learning is often deployed to find malicious behavior in large amounts of data. A rational adversary familiar with this process can easily modify its behavior to avoid detection. In this thesis, the student will:

- 1) Review the existing models using both machine learning and game theory for robust malicious behavior detection.
- 2) Design a new model or extend an existing model to allow for capturing novel aspect of the problem.
- 3) Design an efficient algorithm for computing optimal strategies for the detector in the new model and improve efficiency of algorithms for existing models.
- 4) Experimentally evaluate efficiency of the computed strategies in comparison to existing models and simple baselines.
- 5) Critically evaluate applicability of the proposed model for solving real world problems.

### Bibliography/Sources:


V. Lisy, R. Kessl, and T. Pevny. Randomized operating point selection in adversarial classification. Machine Learning and Knowledge Discovery in Databases, pages 240 - 255, 2014.

F. Ogwueleka. Data mining application in credit-card fraud detection system. Journal of Engineering Science and Technology 6, 6(3):311 - 322, 2011.

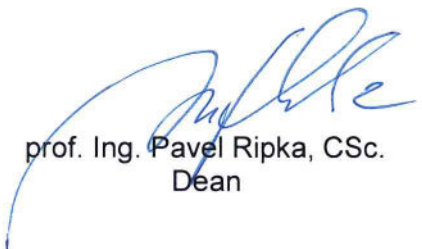
Tambe, Milind. Security and game theory: algorithms, deployed systems, lessons learned. Cambridge University Press, 2011.

Diploma Thesis Supervisor: **Mgr. Viliam Lisý**

Valid until the end of the winter semester of academic year 2017/2018

v. z.   
prof. Dr. Michal Pěchouček, MSc.  
Head of Department



  
prof. Ing. Pavel Ripka, CSc.  
Dean

Prague, April 21, 2016

## **Acknowledgement**

I would like to thank my supervisor Mgr. Viliam Lisý, MSc., Ph.D. for regular and very useful consultations, helpful comments, and for the time Mgr. Viliam Lisý, MSc., Ph.D kindly devoted to controlling my whole work and making it better. Next, I would like to thank Ing. Tomáš Pevný, Ph.D. for regular consultations, valuable pieces of advice and for providing real world data used in the evaluation of the thesis results. I am thankful to my colleagues and managers at O2 Czech Republic Mgr. Jan Svatoš, Ph.D., Mgr. Vít Řádek, Ing. Petr Molinek, Ing. Michal Novák for help in collecting and understanding the data, for explanation of all application-specific details, for valuable discussions and support throughout the course of my work. Last but not least, I would like to thank my family for constant support during my work on the thesis.

## **Declaration**

I declare that I worked out the presented thesis independently and I quoted all used sources of information in accordance with Methodical instructions about ethical principles for writing academic thesis.

Prague, May 26, 2016

.....

## Abstrakt

Klasifikátory založené na strojovém učení se používají v takových bezpečnostních aplikacích jako detekce podvodného chování nebo detekce narušení bezpečnosti počítačových sítí. V aplikacích tohoto druhu klasifikované entity mají tendenci se vyvíjet v čase ve snaze předejít detekci. Avšak klasické metody strojového učení se s tím nedokážou vypořádat jelikož jsou založené na předpokladu že budoucí pozorování budou odpovídat rozložení trenovacích dat. S použitím relevantní literatury možné útoky na klasifikátory vyplývající z této jejich limitace jsou analyzovány spolu s existujícími metodami reakce na nebezpečí útoků. Diskutujeme že místo ignorování existence adaptability útočníků a opravy následně způsobené škody je výhodnější namodelovat soupeře pomocí teorie her s následnou predikcí a omezením jeho možností způsobovat škodu. Jenže byla zjištěna mezera mezi praktickými požadavky na klasifikátory a vlastnostmi existujících modelů herně teoretické optimalizace detekce škodlivého chování. V této práci vyvíjíme postup který vyplní danou mezeru. Praktická aplikovatelnost navržené metody byla vynucena spoluprací s Divizí Bezpečnosti společnosti O2 Czech Republic a.s.: nová metoda byla vyvinuta jako vylepšení interního systému na detekci podvodného chování. Ve výsledku navrženou metodu se dá aplikovat na binární klasifikátor jako na černou skříňku, bez omezení na použitý algoritmus strojového učení. Navíc vyvinutý postup umožňuje omezovat poměr falešných poplachů, což je zásadním požadavkem v bezpečnostních aplikacích strojového učení. Dále model bere v úvahu skutečnost že existují různé typy útočníka. Kromě toho variabilita typů útočníka a zisky v rámci modelu jsou odvozeny na základě nasbíraných datech, což minimalizuje počet hypotéz nepodložených pozorováními. Taky model bere v potaz omezenou racionalitu útočníků. Součástí postupu jsou i vyvinuté efektivní algoritmy na počítání několika herně teoretických konceptů řešení: Nashovy rovnováhy, Stackelbergovy rovnováhy i Stackelbergovy rovnováhy za omezení na poměr falešných poplachů. Díky efektivitě navržených algoritmů je očekáváno že vyvinutý postup zůstane aplikovatelný i pro větší soubory dat. Nakonec efektivita postupu je demonstrována pomocí rozsáhlé experimentální evaluace. S využitím dat z klasifikátorů pro detekce narušení bezpečnosti bylo ukázáno že navržené algoritmy jsou lepší než existující alternativy. Dále na případě detekce podvodného chování v O2 Czech Republic je ukázáno že vyvinutá metoda zachovává účinnost klasifikátoru bez herně teoretické optimalizace na statických datech a vylepšuje robustnost klasifikace pokud se útočník chová v souladu s navrženým modelem.

## Klíčová slova

Teorie her, oponentní strojové učení, algoritmy, bezpečnost, detekce podvodného chování

## Abstract

Machine learning classifiers are used in security applications such as fraud detection or intrusion detection in computer networks. In applications of this kind the classified entities tend to evolve in time attempting to avoid detection. However, classical machine learning methods fail to address the issue, assuming that future observations would follow the same distribution as training data. Based on related work we analyze possible attacks against a classifier arising due to this limitation and survey existing approaches to deal with the attacks. We discuss that rather than ignoring the adaptivity and repairing the damage once it occurs, it is more advantageous to model the adversary by means of game theory and mitigate his ability to cause the damage in a predictive manner. Yet, we identify a gap between practical requirements on adversarial classifiers and properties of the present methods for game theoretic optimization of detecting malicious behavior. In this thesis we develop an approach filling the gap. Practical applicability of the method was enforced by the collaboration with the Security Division of O2 Czech Republic telecommunications company: the novel method was developed as an improvement for the company's internal fraud detection system. As a result, the devised method can be applied to any binary classifier as a black box, not limiting the modeling power of the used machine learning algorithm. Moreover, the approach enables restricting a false alarm rate, satisfying a crucial requirement in the security domain. Furthermore, the model takes into consideration the fact that there are different types of adversaries. In addition, both variability of the adversary types and utilities in the model are derived based on collected data, minimizing hypotheses unfounded with observations. The model also addresses the bounded rationality of adversaries. As part of the approach, we develop efficient algorithms for computation of several solution concepts: Nash equilibrium, Stackelberg equilibrium and Stackelberg equilibrium under the restriction on false alarm rate. Thanks to the algorithms efficiency the approach is expected to remain applicable in case of large datasets. Finally, the efficacy of the developed approach is demonstrated via extensive experimental evaluation. Using data from real-world intrusion detection classifiers, it is shown that the developed algorithms are superior compared to available alternatives. Next, on the case of O2 Czech Republic fraud detection it is demonstrated that the developed method preserves out-of-sample performance of the classifier without the game theoretic optimization, while improving robustness of the classification when the attacker behaves in accordance with the model.

## Keywords

Game theory, adversarial machine learning, algorithms, security, fraud detection

# Contents

|  |           |
|--|-----------|
| <b>1. Introduction</b>   | <b>1</b>  |
| 1.1. Motivation . . . . .  | 1         |
| 1.2. Aim and Outline of the Thesis . . . . .                               | 3         |
| 1.2.1. Goals . . . . .   | 3         |
| 1.2.2. Outline . . . . .   | 5         |
| <b>2. Problem Analysis</b>   | <b>7</b>  |
| 2.1. Taxonomy of Attacks Against Machine Learning Classifiers . . . . .    | 8         |
| 2.1.1. Influence . . . . .   | 8         |
| Causative Attacks . . . . .  | 8         |
| Exploratory Attacks . . . . .  | 9         |
| 2.1.2. Security Violation . . . . .  | 9         |
| Integrity Attacks . . . . .  | 9         |
| Availability Attacks . . . . .   | 9         |
| Privacy Attacks . . . . .  | 9         |
| 2.1.3. Specificity . . . . .   | 9         |
| Targeted Attacks . . . . .   | 10        |
| Indiscriminate Attacks . . . . .   | 10        |
| 2.2. Real-world Problems to Apply Our Results to . . . . .                 | 10        |
| 2.2.1. Machine Learning Module for the O2 CZ Fraud Detection System        | 10        |
| 2.2.2. Addressed Type of Attacks on Machine Learning Classifiers . . .     | 13        |
| 2.3. Adversarial Adaptability and Security by Design . . . . .             | 13        |
| <b>3. Background and Related Work</b>                                      | <b>17</b> |
| 3.1. Basic Game-Theoretic Definitions . . . . .                            | 17        |
| 3.1.1. Normal-Form Game . . . . .  | 17        |
| 3.1.2. Bayesian Game . . . . .   | 18        |
| 3.1.3. Players' Strategies . . . . .                                       | 18        |
| 3.1.4. Solution Concepts . . . . .   | 19        |
| Nash Equilibrium . . . . .   | 19        |
| Bayes–Nash Equilibrium . . . . .   | 19        |
| Stackelberg Equilibrium . . . . .  | 20        |
| 3.2. Receiver Operating Characteristic . . . . .                           | 20        |
| 3.2.1. Classifier Performance Measures . . . . .                           | 21        |
| 3.2.2. ROC curve . . . . .   | 22        |
| 3.3. Related Work on Adversary-aware Classification Models . . . . .       | 23        |
| NE-based Models . . . . .  | 24        |
| SE-based Models . . . . .  | 25        |
| 3.3.1. Categorization of The Models . . . . .                              | 27        |
| 3.3.2. State-of-the-art to Capitalize the Novel Model on . . . . .         | 27        |
| <b>4. Novel Model of Adversarial Classification</b>                        | <b>31</b> |
| 4.1. Bayesian Game of Data-driven Security against Varying Adversary . . . | 32        |
| 4.1.1. Players . . . . .   | 32        |
| 4.1.2. Actions . . . . .   | 32        |
| Defender . . . . .   | 32        |
| Attacker . . . . .   | 33        |

|  |  |           |
|--|--|-----------|
| 4.1.3.   | Player Types . . . . .   | 34        |
| 4.1.4.   | Utilities . . . . .  | 34        |
| Defender . . . . .                                       | 34   |           |
| Attacker . . . . .                                       | 36   |           |
| 4.1.5.   | Bayes-Nash Equilibrium . . . . .   | 36        |
| 4.2.   | Normal-form Game . . . . .   | 37        |
| 4.2.1.   | Utilities . . . . .  | 37        |
| <b>5.</b>  | <b>Efficient Algorithms for Optimal Solutions</b>  | <b>39</b> |
| 5.1.   | General Model Structure . . . . .  | 39        |
| 5.2.   | Nash equilibrium . . . . .   | 40        |
| 5.2.1.   | Set of thresholds . . . . .  | 41        |
| 5.2.2.   | Computation of the players' strategies . . . . .   | 43        |
| 5.3.   | Stackelberg equilibrium . . . . .  | 47        |
| 5.3.1.   | Computation of the players' strategies . . . . .   | 48        |
| 5.4.   | FPR Restriction . . . . .  | 56        |
| 5.4.1.   | SSE computation . . . . .  | 56        |
| <b>6.</b>  | <b>Experimental evaluation</b>   | <b>65</b> |
| 6.1.   | Evaluation of the Algorithms Computation Efficiency . . . . .                              | 65        |
| 6.1.1.   | NE computation . . . . .   | 65        |
| 6.1.2.   | SSE computation . . . . .  | 67        |
| 6.1.3.   | SSE computation under the restriction on FPR . . . . .                                     | 68        |
| 6.2.   | Evaluation of Developed Adversarial Classification on the O2 CZ appli-<br>cation . . . . . | 71        |
| 6.2.1.   | Utilities Derived based on the O2 CZ Data . . . . .  | 72        |
| 6.2.2.   | Performance . . . . .  | 74        |
| Out-of-sample Performance Estimation . . . . .           | 74   |           |
| Performance Estimation on the Altered Datasets . . . . . | 76   |           |
| <b>7.</b>  | <b>Conclusion</b>  | <b>77</b> |
| <b>Appendices</b>  |  |           |
| <b>A.</b>  | <b>Abbreviations</b>   | <b>80</b> |
| <b>B.</b>  | <b>Model Notation</b>  | <b>81</b> |
| <b>C.</b>  | <b>Ideas regarding computation of NE under the restriction on FPR</b>                      | <b>82</b> |
| <b>D.</b>  | <b>The attached CD contents</b>  | <b>85</b> |
| <b>References</b>  |  | <b>86</b> |





# 1. Introduction

“ *If you know the enemy and know yourself, you need not fear the result of a hundred battles. If you know yourself but not the enemy, for every victory gained you will also suffer a defeat.* ”

---

Sun Tzu, *The Art of War*, the 5th century BC

## 1.1. Motivation

Machine learning is presently one of the fastest developing fields of computer science due to the wide, diverse, and practical applications, driven by the availability of data and cheap computational power. Advances in machine learning made it an efficient tool for detection of malicious behavior in a wide variety of security applications such as steganalysis [1], intrusion detection in computer networks [2], spam filtering [3], computer viruses detection [4, 5], biometric personal authentication [6], watermarking [7], and fraud detection [8, 9, 10, 11, 12, 13, 14, 15, 16, 17]. The author of this thesis developed a machine learning classification module for the internal fraud detection system of a telecommunications company O2 Czech Republic a.s.

In contrast with classical machine learning tasks, in security domain the detected objects attempt to undermine the classifier [18, 19, 20, 21, 22, 23, 24, 25, 26]. Adaptivity of adversaries was observed in spam filtering, or fraud detection [27, 11, 21, 13, 28]. It was empirically confirmed that adversaries adapt to existing classification systems: features of malicious instances were found to evolve in time to eventually look more like features of legitimate ones. As a result, adaptive attacks make machine learning detection system less efficient [21].

Classical machine learning methods were found to be vulnerable to adaptive adversaries [29, 30], who degrade performance of security systems [24], causing industry higher financial losses. The average loss due to frauds per company increased approximately by 40 % in 2 years from roughly US \$1.7 million in 2005 to nearly US \$2.4 million in 2007 [31]. Due to fraud the telecommunications industry reportedly used to lose \$2.5 million per year in the late nineties [32] and now reportedly losses more than \$150 million annually [33]. In fraud detection one of the main opened issues, which makes the problem so challenging, is the dynamic adaptivity of frauds [13, 28, 11]. As a result, companies are interested in developing defense systems which are able to preserve their effectiveness under attacks by adaptive adversaries [21].

In the literature there are several possibilities how to improve robustness of the machine learning defense systems against rational adversaries. The most promising one is to develop a predictive defense, because a reactive defense obviously does not suffice and it is necessary to explicitly forecast and preemptively counter possible future attacks [21]. Several approaches were formulated, in which future attacks are being foreseen and incorporated into machine learning defense design by modeling the rational

## 1. Introduction

adversary and his interaction with the security system [34, 21, 24, 35, 36, 37, 38, 39, 40]. The modeling is done using a framework of game theory. A wide variety of studies suggest that game theory is in general an efficient tool for understanding adversary evolution in security [41, 34, 24, 21, 36, 42, 43]. Regarding adversarial classification, it was shown that game theoretic adversary-aware classification models can provide more robust defense systems against rational attackers [35, 24, 21, 36, 37]. The approach of adversary-aware classification, when a designer models strategic interaction between the defense system and the adversary and proactively predicts potential attacks, corresponds to the strong principle of security by design [20].

However, the majority of present adversary-aware classification approaches provide deterministic classification [19]. At the same time, randomization is recognized to be helpful in security applications: it mitigates the capability of adversaries to exploit predictable security patterns to their advantage [44, 42]. There were several successful attempts to incorporate randomization into adversarial classification approaches. In [23] noise was added to the classification boundary of adversary-aware game theoretic classification model from [24]. It was formally shown that even ad-hoc randomization of the decision boundary makes it harder for adversaries to evade the classifier. In [45] it was argued that randomization can be a defense against several types of attacks on machine learning systems. The theoretical results were confirmed in a recent empirical behavior study of spam filter evasion, where it was shown that randomization degrades evasion performance of rational attackers [18]. Even though an ad-hoc randomization improves robustness of adversarial classification, adversary-aware randomization is considered more efficient due to explicit modeling of the adversary. In recent works [36, 35] first adversary-aware randomization of classifier decision boundary was introduced. In [36] it was shown that such randomization forces a rational adversary to design weaker attacks, and in [35] randomized adversary-aware classification was demonstrated to outperform all state-of-the-art deterministic adversary-aware classification methods.

And the main problem with the most of existing adversary-aware approaches is that their application in practical real-world defense systems is either impossible or very limited [21]. Realistic constraints might be considered too complex to be incorporated into game-theoretic approaches [22]. This lead to several limitations of the majority of existing methods [35, 29, 46]:

*a)* most of the methods are unable to account for the real-world operational constraints. To the best author's knowledge the only method incorporating operational constraints is a recent work [35], where an expected number of all processed instances can be restricted. However, there are no methods enabling restriction of the expected false positive rate, which is a primal practical requirement: blocking legitimate cases (e.g., filtering out non-spam messages or disabling services to non-fraudulent customers) often corresponds to critical and dominant costs for companies in real-world settings [35, 24, 13, 39, 29, 47, 28]. It was reported that in practice high false positive rate is an often complaint about systems detecting malicious behavior. Consequently, limiting false positive rate was argued to be a top priority for practical adversarial classification [5].

*b)* Most of the existing methods have limited practical applicability due to high computational complexity.

*c)* Most of the methods are restricted to deterministic decisions not leveraging from effectiveness of randomization in security.

*d)* The majority of the methods do not take real-world variability of adversaries into consideration.

*e)* Most of the methods consider adversaries to be fully rational, while it is more

realistic to model partial rationality of the attackers.

f) The majority of existing methods put significant restrictions on the machine learning method (e.g., linear or logistic loss function).

These challenges must be addressed, applicable in practice adversary-aware security is required, as attacks by rational adversaries are becoming more and more sophisticated [26]. In recent years cyber attackers are becoming increasingly more rational [48, 37]. Examples of successful attacks by highly sophisticated adversaries can be spear phishing attacks on the RSA company [49], on the Oak Ridge national security and energy laboratory, White House and the Nuclear Regulatory Commission of the U.S. [47], or attacks by the Stuxnet worm which reportedly ruined several nuclear centrifuges, bypassing security and increasing the centrifuges pressure, while sending false measurements data to the control room [34].

## 1.2. Aim and Outline of the Thesis

Adversarial machine learning is an emerging field of research. The aim of this work is to develop a novel applicable adversarial machine learning approach which can be directly deployed in such real-world applications as the O2 Czech Republic fraud detection system. The requirements on a new method are: it must

- a) be highly accurate,
- b) enable control over expected false positives rate,
- c) be robust against detection avoidance by adaptive adversaries,
- d) model a real-world variability of rational adversaries,
- e) enable to set an application-specific adaptability rate of the adversaries,
- f) create a realistic model, based on historical data,
- g) be computationally efficient.

### 1.2.1. Goals

The listed goals **a) - g)** are to described in more details.

#### **a) High accuracy**

In order for the approach to be highly accurate, it must enable the usage of any state-of-the-art machine learning method as a black box without any restrictions both on the algorithm and on training data, unlike the majority of existing adversarial classification models which either directly incorporate the classification method into the model or require significant restrictions on the learning algorithms or on training data.

#### **b) Control over expected false positives rate**

The approach must enable control over expected false positive rate of the final system, in order to make it possible to deploy the approach in a real-world setting. Unbounded false positive rate was pointed out as one of the major limitations for deployment of adversarial classification in practice [5]. Still, to the best author's knowledge there is no adversary-aware classification model optimizing performance of the security system under a specified restriction on the total expected false positives rate. The only adversary-aware model involving operational constraints was developed in the recent work [35, 50]. However, in the model due to [35, 50] a total expected number of all

## 1. Introduction

alarms, rather than false alarm rate, is restricted. In the case of the O2 CZ fraud classifier, as well as in many other practical applications, it is operationally possible and is required to detect as many malicious cases as possible. However, high false alarm rate results in the denial of service in practical adversarial classification [5, 35, 24, 13, 39, 29, 47, 28]. Thus, the goal is to address a problem in the style of the Neyman-Pearson approach to decision theory [51, 52]: the novel approach must improve the robustness of security under the constraint on expected false positive rate.

### **c) Robustness against detection avoidance by adaptive adversaries**

In order for the approach to be robust against adaptive adversaries, rationality of adversaries and their interaction with the defense system must be modeled using a formal mathematical framework: game theory.

Furthermore, unlike most of the existing game theoretic models which stick to deterministic classification [35], the novel approach must enable leveraging from randomization which was shown to make detection avoidance harder for adversaries [18, 45].

### **d) Variability of rational adversaries**

The approach must address the real-world fact that rational adversaries are not identical. To the best author's knowledge, [35] is the only model taking realistic variability of adversaries into consideration. Moreover, the model from [35] addresses goals **e)** and **f)** as well.

### **e) Adaptability rate of the adversaries**

The novel approach must enable setting an application-specific adaptability rate of the adversaries. It is unrealistic to assume all malicious entities to be fully rational. In real-world cases adversarial rationality is bounded. The approach must enable setting an application-specific expected probability that an adversary is adaptive. This would enable the usage of the approach in both applications with highly adaptable attackers, and in applications where rationality and, thus, adaptability of the adversaries are rather bounded.

### **f) Data-driven modeling**

The majority of existing game-theoretic models for adversarial classification are based on idealized modeling of utilities, which limits practical applicability of the models [35]. Modeling the utilities and variability of the adversaries in the approach must be driven directly by the real-world data, aiming to avoid any unrealistic assumptions.

### **g) Computational efficiency**

Last but not least, as a part of the new approach, efficient algorithms for computation of the model solution must be developed. Expressive models tend to lead to intractable in practice formulations. For instance, the model from [35], which successfully addresses goals **c)** - **f)** and moreover enables usage of arbitrary machine learning method, results in an infinitely large LP formulation if at least one attribute in training data is continuous.

### 1.2.2. Outline

The rest of the thesis is structured as follows.

In Chapter 2 we analyze which types of attacks against a machine learning classifier are possible. A general taxonomy of the attacks is provided. Next, we describe the O2 CZ fraud detection application and specify the type of attacks we focus on. Finally the chapter is concluded with analysis of possible ways to deal with the attacks and identification of the most promising ones. In Chapter 3 we overview the background required to deal with the promising approaches and then we analyze existing methods from the perspective of the practical requirements stated as goals in Section 1.2.1. Capitalizing on the most promising approaches from Chapter 3, in Chapter 4 we develop a novel game theoretic model of adversarial classification addressing several research goals. Next, in Chapter 5 we analyze the formulated model and discover several facts about it. Based on the discoveries, we devise efficient algorithms for computation of several game theoretic solution concepts in the model. In Chapter 6 we demonstrate results of the extensive evaluation of the proposed methods. Finally, in Chapter 7 we conclude this work.

## *1. Introduction*

## 2. Problem Analysis

In the previous chapter we have introduced an emerging field of adversarial machine learning, discussed why it is crucially important for the industry to make state-of-the-art classification robust against avoidance attacks and what are the main opened problems in the field which we aim to address. In this chapter we first provide a general categorization of existing types of attacks on machine learning classifiers. Next, in terms of a general taxonomy we categorize a broad type of avoidance attacks this work focuses on. This is the most common and the most frequently studied type of adversarial problem arising in security applications of machine learning [26]. Therefore, the reader should be able to directly use the results of this work in many real-world applications of adversarial classification in order to improve its robustness and, consequently, long-term performance. In order to keep the description illustrative and all assumptions in the following chapters realistic, the addressed type of the real-world problems is illustrated with the fraud detection module for the O2 Czech Republic company, a.s. We conclude the chapter with discussion why for the specified problem a proactive adversary-aware classification approach is required, comparing it to a currently common reactive approach of dealing with the problem once it occurs [21, 26, 20].

### Rational Adversary

In the following taxonomy we frequently use a term *rational adversary* who attacks the machine learning classifier. In order to avoid ambiguity and to improve the reader's experience, we are about to briefly discuss what is meant by *the rational adversary* in the following section.

In applications of adversarial machine learning classified entities are rational<sup>1</sup>. The goal of the classifier designer is to distinguish malicious and legitimate instances apart. At the same time malicious rational entities have intention to avoid detection, in other words to be misclassified as legitimate instances. Rational malicious entities tend to analyze how close their behavior is to possible legitimate behavior. They tend to analyze what are the odds that a security system would classify them to be malicious. Based on such reasoning they decide how much they should modify their behavior in order to remain undetected. Cumulative behavior of rational malicious entities cause a decrease in the security classifier performance. In the literature this cumulative behavior is considered to be an attack on the machine learning system. The attack is modeled as an action of a rational adversary who has intention to increase the overall number of false negatives<sup>2</sup>[54, 34, 47, 37, 48, 35, 22, 55, 25, 24]. In some security applications, like fraud detection, the rational adversary might be intuitively considered as some criminal mastermind who coordinates malicious entities. In other security applications several instances can be produced by a single person, spam detection is an example of such applications. In the later case the rational adversary can be viewed as an author of all spam messages. In addition, introducing the concept of the rational adversary provides

---

<sup>1</sup>In general a decision-maker is *rational* if he consistently pursues his own objectives [53].

<sup>2</sup>False negatives are malicious instances misclassified as legitimate. Analogously, false positives are legitimate instances wrongly classified as malicious. For more details see Section 3.2.

certain modeling flexibility. Using the concept, it is possible to address cases when a legitimate instance can be generated by the rational adversary, as well as cases when the rational adversary not only adapts to the security classifier but also attempts to corrupt the classifier before its deployment by providing misleading training data.

In the following, the term *rational adversary* would have the discussed meaning, if explicitly not stated otherwise. Moreover, terms *the adversary* and *the attacker* would be used interchangeably.

### 2.1. Taxonomy of Attacks Against Machine Learning Classifiers

A qualitative taxonomy of possible attacks by an adversary against machine learning classifiers was first introduced in [45] and then the taxonomy was extended in [26]. The taxonomy was used in design of frameworks for empirical evaluation of a classifier security under adversarial attacks [56, 20]. Even though the field of adversarial machine learning is relatively young, it already deals with a broad variety of real-world problems. The taxonomy serves as a formal map to orientate in the diverse world of adversarial machine learning problems. It will help us to formally specify the subject of this thesis and differentiate it from related yet different problems.

The taxonomy differentiates all possible attacks based on several properties: *a)* the type of *influence* that a rational adversary has on the classifier, *b)* the type of *security violation* the undetected attack causes, and *c)* the level of *specificity* of the attack. According to the taxonomy, each attack against a classifier is categorized with a combination of values for each property. Intuitively, the category of each attack is a point in a three-dimensional space, where the axes are *a)* the *influence*, *b)* the *security violation*, and *c)* the *specificity* of the attack. Possible values for each one of these attack properties will be described.

#### 2.1.1. Influence

This property characterizes the capability of the adversary to influence the classifier. Based on the type of attacker’s influence, the attacks are divided into *a)* *causative*, and *b)* *exploratory*.

#### Causative Attacks

In case of causative attacks the rational adversary has a capability to modify training data which are later used to develop a machine learning classifier. Depending on the application, the adversary might have control over some fraction of training data, or over the whole training dataset. When the adversary controls a fraction of training data, he might control only malicious training samples, or both legitimate and malicious instances. Furthermore, even if all instances from the training dataset are under the adversary’s control, it might be the case that the attacker can manipulate just some aspects of data: for instance, modify just a restricted subset of instance features. Regardless of the described application-specific nuances of causative attacks, the main adversary’s strategy remains the same: first, influence training data in such a way that the produced classifier would have vulnerabilities, and subsequently take advantage of the vulnerabilities once the bad classifier is deployed. For instance, it might be possible for the attacker to perform a causative attack if the classifier is being retrained on-line, i.e. once new data sample arrives, and if the adversary is able to produce sufficient



amount of modified instances [26, 57]. Note that in case of causative attacks the adversary might be able to modify the testing data as well as the training data. However, the idea of this type of attacks is to harm a classifier during the training phase in such a way, that later malicious instances would be wrongly classified as legitimate because of the initially corrupted classifier.

### Exploratory Attacks

In case of exploratory attacks the rational adversary attempts to bypass the already deployed machine learning classifier without previous modifications of the learning process. In attacks of this type the rational adversary tries to make the malicious samples to be viewed by the deployed security system as legitimate. Depending on application-specific details, the adversary might have control either over all malicious instances or over some subset of the instances. Analogously to the case of causative attacks, it might be the case that the adversary can do any modifications of a controlled instance, being able to change any attributes of the sample. It also might be the case that the adversary's control over instances is limited, for instance, it could be possible for the attacker to modify just a subset of the attributes.

#### 2.1.2. Security Violation

Based on the type of the caused security violation, there are three sorts of attacks: *a) integrity*, *b) availability*, and *c) privacy* attacks.

### Integrity Attacks

In this type of attacks the adversary aims to bypass the deployed classifier. Such attacks are closely related to exploratory attacks. In [26] the whole section on exploratory attacks is dedicated to exploratory integrity attacks. At the same time the attacker might perform the causative integrity attack by modifying training data. In case of integrity attacks the adversary's goal is to increase a number of false negatives of the machine learning classifier. Increasing a number of false negatives is considered the main realistic goal of the adversary attacking a machine learning system [54, 34, 47, 37, 48, 35, 22, 55, 25, 24].

### Availability Attacks

In this type of attacks the attacker tries to make the system effectively unusable. By means of either causative or exploratory attacks the adversary attempts to degrade the overall classifier performance, so that the security system would be shut down. In order to cause the denial of service, the attacker intends to increase both a number of false negatives and a number of false positives.

### Privacy Attacks

In case of privacy violation attacks, the adversary probes the classifier trying to reveal the confidential information about the used training instances.

#### 2.1.3. Specificity

The last property describes to what extent the adversary's intentions are specific. There are *a) targeted* and *b) indiscriminate attacks*.

### Targeted Attacks

In case of targeted attacks the adversary desires several particular malicious instances to bypass the classifier.

### Indiscriminate Attacks

In case of indiscriminate attacks the adversary pursues a more general goal of degrading the classifier's performance on a broader set of malicious instances. The attacker's goal in case of indiscriminate attacks can be formulated as producing any false negative.

## 2.2. Real-world Problems to Apply Our Results to

One of the main motivations for this work is to develop and deploy a robust and efficient machine learning classifier for a fraud detection system in the O2 Czech Republic company. At the same time practical limitations of current state-of-the-art models for adversarial classification make their deployment in production environment either impossible or unreasonable. Addressing the opened issues of adversarial machine-learning in a context of a real-world application forces the author to keep all assumptions of formal modeling realistic and to resolve relevant issues arising in practice. Moreover, a real-world use case would make dealing with formal modeling more illustrative. For those purposes, an adversarial classification case of the O2 CZ fraud detection system is briefly introduced.

The O2 CZ case description should help the reader to better understand a type of real-world applications the results of this work can be directly applied to. The addressed problem represents a general type of the adversary-related problem commonly arising in security applications of machine learning [26]. The studied problem will be analyzed and categorized from the perspective of the general attacks taxonomy.

### 2.2.1. Machine Learning Module for the O2 CZ Fraud Detection System

Using state-of-the-art machine learning practices [58], a new module for the internal fraud detection system at O2 Czech Republic has been designed and implemented.

The existing internal fraud detection system is quite complex. One of core subsystems detects fraudulent customers of the company who actively use various services but intend not to pay for it. Let us call this subsystem *a subscription-fraud system*<sup>3</sup>. The existing complex subscription-fraud system consists of various previously developed modules which are used by operators of the O2 CZ fraud division. Even though the existing subscription-fraud system has proven himself to be efficient, some fraudulent customers still manage to bypass the system. It is primarily given by the adaptability of frauds [13, 28, 11]. The author of this work focused on developing the new last line of defense for the existing subscription-fraud system as a part of his O2 CZ student internship. For this purpose one-year data on instances inspected by the system and discarded as legitimate were collected. The collected data were labeled based on the fact if the inspected but discarded customer eventually payed for the service. After understanding, preprocessing the data and reviewing current state-of-the-art in the detection of fraud by means of supervised machine learning, the author developed a predictive model. The company data analysts and management were satisfied with performance of

---

<sup>3</sup>The meaning of a subscription fraud will be briefly explained in the following text.

the developed prototype and it was decided to invest in further improvements and deployment of the last-defense classification module into the production environment. As the developed model is meant to be the last line of defense against the most non-trivial fraudulent cases, it was required to increase robustness of the module against rational adaptability of fraud. However, no current state-of-the-art method of adversary-aware classification satisfied all the practical requirements listed in Section 1.2. As a result, the goal of this thesis was formulated to improve state-of-the-art game-theoretic optimization of adversarial classification making it applicable in practical settings. Even though neither the machine learning classifier itself nor telecommunications frauds in general is a subject of this work, both will be briefly introduced to the reader. Dealing with a real-world application throughout the thesis will keep the work illustrative. Moreover, the following introduction of the underlying classifier together with non-restrictive requirements on it will help the reader to better understand which type of adversarial classification applications would leverage from the results of this work.

The fraud detection module introduction is structured as follows. First, it is mentioned what is often meant by telecommunications fraud and what is the primary goal of the existing internal subscription-fraud detection system. Next, general properties and requirements on the underlying classifier are discussed. Finally, the dataset used for developing the classifier is introduced, so that when throughout this work the author would refer to the O2 CZ dataset, the reader would be comfortable with following the discussion.

### **Addressed Telecommunications Fraud**

Telecommunications fraud can be in the most general form defined as any activity by which service is obtained without intention of paying [59, 32]. More specifically, the addressed type of frauds is some times called *contractual fraud* which generates revenue through the unforbidden use of the service with no intention of paying for the use. The contractual fraud can be divided into *subscription fraud* and *premium rate fraud* [32]. The latter type occurs when fraudsters increase a number and/or duration of calls to a premium rate number, the phones which have been calling the premium rate number might eventually not pay their high bills [60, 32]. Subscription fraud can be divided into two classes: one includes new customers who sign up for the service with no desire to pay, in this type of the subscription fraud false identification can take place, while another class of subscription fraud includes customers who decide part way through their contract that they would no longer pay for the use of the service [33, 32]. The subscription fraud was argued to be the most common type of fraud encountered by telecommunications companies [10]. Detection of subscription fraud can be considered the primary goal of the existing fraud detection system.

### **Underlying Classifier**

For the purpose of this work it is sufficient to know that the last-defense module was created using labeled historical data and methods of supervised machine learning. In general, the goal of malicious behavior detection is to classify a previously unseen instance as either malicious or legitimate. Analogously, the developed machine learning module classifies a previously unseen user as either fraudulent or benign. In other words, the underlying machine learning method performs binary classification. Out of several developed classifiers the one with the best performance was selected. The classifiers'

## 2. Problem Analysis

performance was mainly evaluated using ROC curves<sup>4</sup>. The chosen machine learning methods for design of the classifiers were either considered state-of-the-art supervised approaches in the applied fraud detection research [8, 13, 10, 14, 15] or/and have proven themselves to be very efficient achieving state-of-the-art performance in machine learning challenges, e.g. competitions on the Kaggle platform: <http://www.kaggle.com>. However, practically any supervised machine learning classifier, or classifier ensemble [61], can be used as a black box in the developed method of efficient adversary-aware classification. The only restriction on the resulting binary classifier is to enable producing an ROC curve<sup>4</sup>, which is a broadly used technique for evaluation of binary classifiers performance. To put it differently, for the classifier it must be possible to determine the degree to which a particular instance belongs to a positive class. Note that in practice it is not a restrictive requirement. The majority of state-of-the-art classification models in detection of malicious behavior naturally output a continuous quantity, a probability value or a score which represents the degree to which an instance is a member of a positive class, and final classification is made by thresholding the produced quantity [8, 13, 10, 14, 15]. Examples of the classifiers which output continuous scoring quantity might be: Naive Bayes classifier, neural networks and their ensembles, ensembles of classification and regression decision trees, such as random forests and boosted trees<sup>5</sup>. In addition, implementations of some popular machine learning models outputting discrete labels, such as Support Vector Machines, directly provide an option of predicting the class probabilities<sup>6</sup>.

To sum up, based on state-of-the-art machine learning practices and using labeled data the machine learning classifier was developed. A strict requirement on adversary-aware method improving the long-term robustness was to not restrict the used machine learning method in any way, e.g., by restricting its decision boundary or its loss function, see the goal **a**) in Section 1.2.1. The requirement was to take the developed classifier and improve it for the adversarial setting. Besides not imposing any restrictions on the machine learning method, it was also required to not lose any information available in data. Thus, no assumptions about the data were allowed. The used labeled dataset is to be briefly described.

### Collected Dataset

The dataset used to develop the machine learning classifier contained 20 484 of historical instances. All instances were labeled based on information whether or not a person eventually paid for the used services. As one would expect based on the previous description of the challenging nature of the last-defense problem, class distribution of the dataset was significantly skewed: there were just 1941 positive (fraudulent) cases. Each instance was described using a set of features one would naturally expect for this kind of applications. Among others, there were features equal to amount of money spent by a person during previous periods of time, e.g. during the last 24 hours and during the last 28 days. The money-based features were represented by floating-point

---

<sup>4</sup>See Section 3.2 for more details on an ROC curve.

<sup>5</sup>It is worth noting that recently an incredibly efficient implementation of gradient tree boosting algorithm appeared: a scalable tree boosting system named XGBoost [62]. The system has already proven itself very useful achieving state-of-the-art results in various machine learning challenges, e.g., competitions on the Kaggle platform: <http://www.kaggle.com>. Moreover, XGBoost implementation "runs more than ten times faster than existing popular solutions on a single machine and scales to billions of examples in distributed or memory-limited settings"[62].

<sup>6</sup>E.g., in libsvm-based R package for Support Vector Machines, "the probability model for classification fits a logistic distribution using maximum likelihood to the decision values of a binary classifier"[63].

numbers with two decimal places. Some features contained information about a person's history, e.g. for how long the user is a company's customer, statistics on payments during the contract, etc. Some features captured the pattern of the person's behavior, e.g. what services he used lately. There were as well features capturing demographic characteristics of a customer: for instance, a reported by the customer age category, or a safety score for a place where the person lives, etc. The dataset contains more features, including several engineered ones, however, for the purpose of the illustration the mentioned features would suffice. The overall number of features used for the final model training was eighteen. As it might be obvious from the overview, in the dataset there were both features with continuous domains and discrete-domain features. Note that in the following text terms feature and attribute are used interchangeably.

#### 2.2.2. Addressed Type of Attacks on Machine Learning Classifiers

We have discussed the O2 CZ fraud detection case, which motivated this research. Now, the studied type of the avoidance attack against supervised machine learning classifiers is to be categorized in terms of the general taxonomy.

Regarding the capabilities of the adversary, which is the first attack property in the taxonomy, in this work we target exploratory attacks.

In [64] it was discussed that in many applications causative attacks are too imprecise and of a limited practical use to the adversary: in order to produce a significant fraction of training data, it requires large amount of time, powerful computational resources, and/or system privileges. This is obvious for the O2 CZ fraud detection as well: in order to produce a single positive training instance, a person has to sign up for the service, actively use it in the same manner as if he plans not to pay for the service, and in the end of month pay the bill. Still, without devising information about the deployed classifier, the causative attack itself does not provide the adversary any guarantees for not being detected [64].

In terms of the caused security violation, we address integrity attacks. Exploratory integrity attacks against machine learning systems are the most common in practice [26]. It is often considered the case that the rational adversary is mainly concerned with causing as many false negatives of the system, as possible [54, 34, 47, 37, 48, 35, 55, 25, 24, 22].

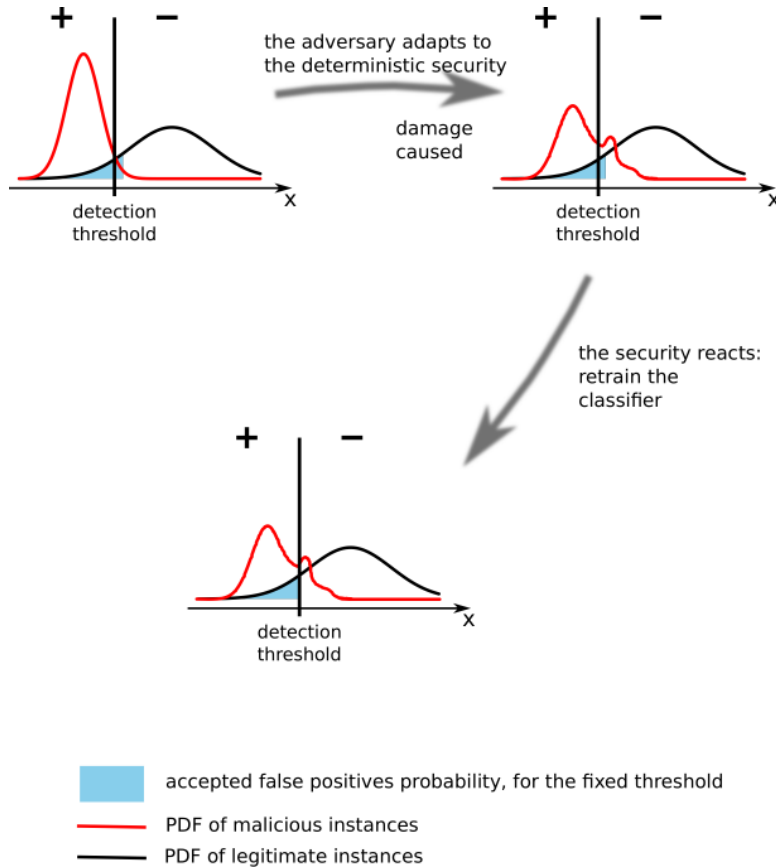
Regarding the type of the caused security violation, the results of this work can be applied both in case of targeted and in case of indiscriminate attacks, because the novel approach enables to derive types of attacks from a collected case-specific data. In the case of the O2 CZ fraud detection, the attack can be better described as an indiscriminate one.

### 2.3. Adversarial Adaptability and Security by Design

To conclude the problem analysis, we will examine possible approaches how to deal with adaptability of adversaries in security classification. It will be shown that in machine learning applications in security domain attacks by the adaptive adversary should be taken into consideration during the design phase, in order to build a system secure from the ground-up.

One of the main open issues in fraud detection is the dynamic adaptability of frauds [13, 28, 11]. Adversarial adaptability was observed in other security applications of machine learning as well [27, 21]. However, very often in practice the issue is not considered during the design phase. Machine learning classifiers are designed as if the

## 2. Problem Analysis



**Fig. 2.3.1.** Illustration of a reactive security approach

classification task does not deal with rational entities. As the rationality of malicious entities is neglected during the design phase, later it results in decrease of the classifier performance. Once the problem occurs, the classifier designer reacts to it by retraining the classifier on new data [21, 26, 20]. Such a reactive approach to the adversarial adaptability is illustrated in Fig. 2.3.1 on a simple example of classification based on a single attribute  $x$ .

For illustrative purposes, in the depicted example initial populations of both malicious and legitimate instances are assumed to be normally distributed. Based on initial training data the defender develops and deploys an optimal linear threshold-based classifier. However, once some of rational adversaries become aware of the classifier existence, they attempt to avoid detection, changing a shape of the malicious class distribution. It in its turn results in decay of the classifier performance. Once the decay is so considerable that it cannot be ignored any more, the defender collects new data depicting the current state of affairs and reactively retrain the classifier. Despite the simplicity of the example, it also illustrates a constraint on maximum false positive rate, which is crucial for real-world adversarial classification [5, 35, 24, 13, 39, 29, 47, 28]. Due to the constraint on maximal acceptable probability of false alarms, the defender cannot perform an optimal detection of malicious instances any more. To sum up, the occurred problem, which was neglected during the design phase, is so serious that the reactive approach is unable to resolve it completely.

Reactive security arises primarily due to the designer's expectation that details of the deployed system can be kept secret from the rational adversary, this common ap-

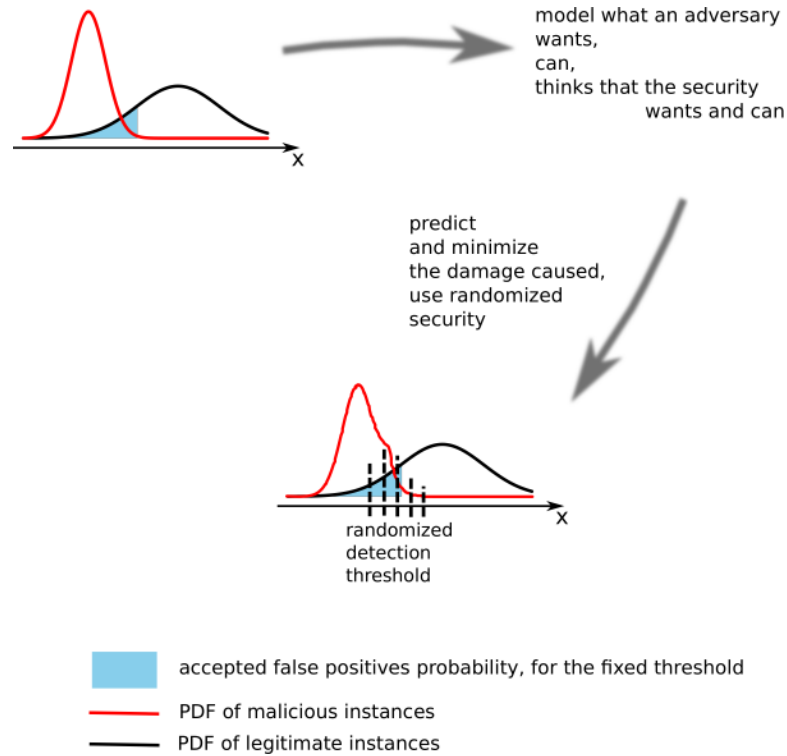


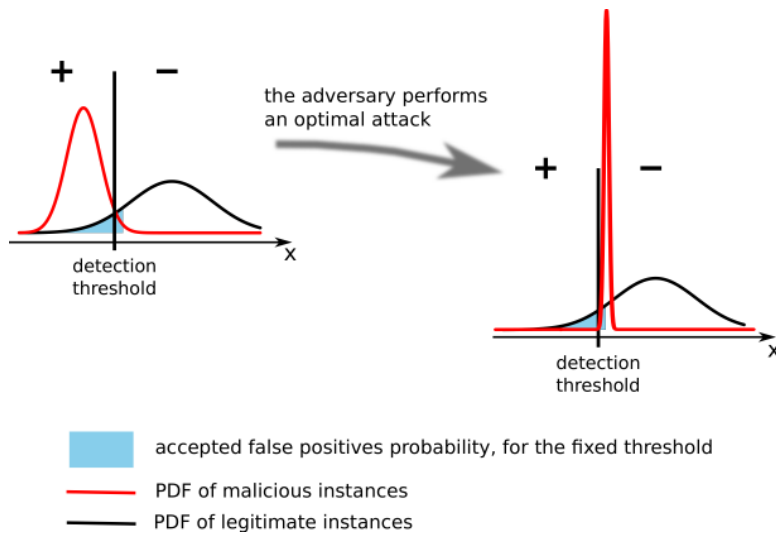
Fig. 2.3.2. Illustration of a proactive adversary-aware approach

proach in engineering is known as *security by obscurity* [20]. However, in case of widely used security applications, such as spam, intrusion, or fraud detection, the latest advances in the applied machine learning techniques are being openly published [8] and application-specific state-of-the-art algorithms are common knowledge [45]. Except for some very rare security applications with a small number of deployments, it is unrealistic even to expect the set of possible instance features to be unknown to the adversary [45]. Furthermore, recently it was formally shown that an arbitrary classifier can be reversed-engineered under real-world restrictions [19]. Additionally, it was proven that any randomized combination of classifiers can be also reverse engineered, though imperfectly [19].

As a result, in order for the system to be secure, a system designer should not rely on unrealistic beliefs in secrecy. Such approach follows the Kerckhoffs's Principle or the paradigm of *security by design* [34, 26, 20]. It was shown that if adversarial adaptability is taken into consideration during the design phase, then the resulting adversary-aware security systems lower the systems losses due to rationality of adversaries [36, 35, 24, 21, 37]. Adversary-aware approach attempts to predict and prevent the problem before it actually occurs, in a proactive manner, see Fig. 2.3.2.

In the figure we continue the described simple example. Fig. 2.3.2 illustrates, that when the problem is anticipated before it actually occurs, it can be possible to minimize the extent of the problem. To put it differently, by understanding intentions and capabilities of the adversary and by deploying adversary-aware randomized classification, losses of a security system can be minimized compared to eventual losses when the adaptability is ignored during the design phase [36]. The Fig. 2.3.2 also illustrates, that when using a randomized classification it is possible to seldom use a classification corresponding to higher false alarm rates without violation of the constraint on maximal

## 2. Problem Analysis



**Fig. 2.3.3.** Illustration of an unrealistic optimal attack

acceptable probability of false alarms. It reduces the ability of adversaries to exploit security patterns to their advantage compared to the case of ignoring the adversarial adaptability. Note that the reactive approach to the problem, see Fig 2.3.1, is still better than the approach of ignoring the very existence of the problem.

To sum up, in order to design a system which would be secure from the ground-up, the ability of the adversary to discover information on the used algorithms, on training data, and on the final classifier must not be ignored. Neither abilities of the adversary should be overestimated, however. It would be unrealistic to assume that the adversary is capable of performing an optimal attack as in Fig. 2.3.3. Therefore, it is required to tune the level of the adversarial adaptability based on a specific application. The approach developed in this work enables that.



## 3. Background and Related Work

In this chapter we overview and systematically categorize existing game theoretic adversary-aware models for robust adversarial machine learning. The categorization is based on real-world requirements of the model applicability, dictated by the O2 CZ fraud detection use case. Before diving into related work, we provide a reader with an overview of background required for understanding this thesis.

### 3.1. Basic Game-Theoretic Definitions

Game theory is a mathematical study of interaction between several rational entities, termed players, such that each player maximizes his own utility. Informally, game theory can be viewed as a tool which helps to optimize a payoff in some dynamic environment, where the dynamic nature of the environment arises due to the fact that all other inhabitants of the environment maximize their own payoffs. Knowing about the discussed adaptivity of the rational adversary, game theory is exactly what is needed to minimize the defender's losses by modeling what the adversary wants, can, and has intention to do when he knows what the security system designer wants and can do, see Fig. 2.3.1. Successful real-world applications of game theory demonstrate that it is in general an efficient tool for modeling and understanding adversarial interaction in security [42, 43].

Note that in this work a framework of non-cooperative game theory is used, which is notably suitable for the studied problem of exploratory integrity attacks. Cooperative game theory is not within the scope of this thesis. If we need to refer to cooperative game theory it would be stated explicitly, otherwise in the following text non-cooperative game theory is meant by *game theory*.

#### 3.1.1. Normal-Form Game

**Definition 3.1.1.** *"The simplest game definition, known as a normal form, contains a tuple  $(N, A, u)$ , where:*

- $N$  is a finite set of  $n$  players who are taking the action.
- $A = A_1 \times A_2 \times \dots \times A_n$  and  $A_i$  is a set of actions available to the  $i^{\text{th}}$  player.
- $u = (u_1, u_2, \dots, u_n)$  and  $u_i : A \mapsto \mathbb{R}$  is a utility function of the  $i^{\text{th}}$  player. The utility function of a player maps every vector of actions  $a \in A$  to a real-valued number. This number characterizes a degree of the player's happiness about the outcome of the vector of actions  $a$ , which is also called an action profile. The greater the  $u_i(a)$  is, the more satisfied the  $i^{\text{th}}$  player is about the outcome of the action profile  $a$  ([65], p. 118)."

A normal-form game is often viewed as the most fundamental one. Normal-form game is also called a single-stage game to underline the fact, that the game has no temporal structure. For some problems other game formulations might be more appropriate.

### 3.1.2. Bayesian Game

In case of the normal-form game it is assumed that all players know against whom they play. However, for some real-world settings it required to model the player's uncertainties about an adversary. This is possible using a model called *Bayesian game*. The uncertainty is often modeled as a probability distribution over possible types of players. We provide a definition of Bayesian game from ([65], p. 167)

**Definition 3.1.2.** *A Bayesian game is a tuple  $(N, A, \Theta, p, u)$ , where:*

- $N$  is a set of players.
- $A = A_1 \times A_2 \times \dots \times A_n$  and  $A_i$  is a set of actions available to the  $i^{\text{th}}$  player.
- $\Theta = \Theta_1 \times \Theta_2 \times \dots \times \Theta_n$ , where  $\Theta_i$  is the type space of  $i^{\text{th}}$  player.
- $p : \Theta \mapsto [0, 1]$  is a common prior over types.
- $u = (u_1, u_2, \dots, u_n)$  and  $u_i : A \times \Theta \mapsto \mathbb{R}$  is a utility function of the  $i^{\text{th}}$  player.

### 3.1.3. Players' Strategies

In game theory a set of all available player's choices is called a set of his strategies. One type of strategies is a pure strategy [65]. For a normal-form game it means that a player chooses one action and plays it deterministically. For a Bayesian game a pure strategy  $a_i$  of the  $i^{\text{th}}$  player is defined as a mapping from each possible type a player might have to his action he performs in case he has this type,  $\alpha_i : \Theta_i \mapsto A_i$  ([65], p. 168).

A selection of an action for every player is called a pure-strategy profile.

We have already discussed the pure strategy. Another type of strategies is a mixed strategy. The definition of mixed strategies from ([65], p. 59) follows.

**Definition 3.1.3.** *"Let  $S_i$  be a set of all pure strategies the  $i^{\text{th}}$  player has, and for any set  $X$  let  $\Pi(X)$  be the set of all probability distributions over  $X$ . Then the set of mixed strategies for the player is  $\Pi(S_i)$ . The set of mixed-strategy profiles is simply the Cartesian product of the individual mixed-strategy sets."*

Pure strategies which are played with non-zero probability form a *support* of the mixed strategy.

It can be shown, that some strategies would be always preferred by a rational player over other strategies, because the later always provide either worse or not better outcome for the player. Such unappealing strategies are termed as *dominated*. Intuitively, a strategy is dominated if there is another strategy which always provides a player either better or equally good outcome, no matter what all other players play. Domination of strategies is not restricted to pure strategies only. If a rational player is always better off when playing one mixed strategy instead of the other, we can describe this case by means of the domination as well. The formal definitions regarding strategy domination from [65] follow.

**Definition 3.1.4.** *"Let  $s_i$  and  $s'_i$  be two strategies of player  $i$ , and  $S_{-i}$  be the set of all strategy profiles of the remaining players. Then*

1.  $s_i$  strictly dominates  $s'_i$  if for all  $\forall s_{-i} \in S_{-i}$ , it is the case that  $u_i(s_i, s_{-i}) > u_i(s'_i, s_{-i})$ .

2.  $s_i$  weakly dominates  $s'_i$  if for all  $\forall s_{-i} \in S_{-i}$ , it is the case that  $u_i(s_i, s_{-i}) \geq u_i(s'_i, s_{-i})$  and for at least one  $s_{-i} \in S_{-i}$ , it is the case that  $u_i(s_i, s_{-i}) > u_i(s'_i, s_{-i})$ .
3.  $s_i$  very weakly dominates  $s'_i$  if for all  $\forall s_{-i} \in S_{-i}$ , it is the case that  $u_i(s_i, s_{-i}) \geq u_i(s'_i, s_{-i})$ .

**Definition 3.1.5.** "A strategy is strictly (resp., weakly; very weakly) dominant for a player if it strictly (weakly; very weakly) dominates any other strategy for that player."

**Definition 3.1.6.** "A strategy  $s_i$  is strictly (weakly; very weakly) dominated for a player  $i$  if some other strategy  $s'_i$  strictly (weakly; very weakly) dominates  $s_i$ ."

### 3.1.4. Solution Concepts

#### Nash Equilibrium

If we know strategies of all players except the  $i^{\text{th}}$  one, assuming that the players are rational, it is possible to predict a strategy of the  $i^{\text{th}}$  player. The  $i^{\text{th}}$  player would choose such a strategy which maximizes his own utility. Such a maximizing strategy as a response to a given set of other players' strategies is called a *best response* of the  $i^{\text{th}}$  player [65]. One of the most powerful and stable solution concepts in game theory is Nash Equilibrium which can be defined using the best response definition. The definitions from ([65], p. 62) follow.

**Definition 3.1.7.** "Let  $S_i$  be a set of possible strategies for the  $i^{\text{th}}$  player. Let  $s_{-i} = (s_1, \dots, s_{i-1}, s_{i+1}, \dots, s_n)$  be a strategy profile without a strategy of the  $i^{\text{th}}$  player. The  $i^{\text{th}}$  player's best response to the  $s_{-i}$  is a strategy  $s_i^* \in S_i$  such that  $u_i(s_i^*, s_{-i}) \geq u_i(s_i, s_{-i})$  for all strategies  $s_i \in S_i$ ."

**Definition 3.1.8.** "A strategy profile  $s = (s_1, \dots, s_n)$  is a Nash Equilibrium (NE) if, for all agents  $i$ ,  $s_i$  is a best response to  $s_{-i}$ ."

#### Bayes–Nash Equilibrium

Compared with a normal-form game, in a Bayesian game there are additional sources of uncertainty. In order to define the best response, it is first required to specify a meaningful notion of the agent's utility. There are several definitions of expected utility for Bayesian games. In this work we will need a so called *ex interim* expected utility, which models the setting when a player is aware of his own type but does not know the types of the other players. The following definitions are taken from ([65], p. 168 - 170).

**Definition 3.1.9.** *Ex interim expected utility of the  $i^{\text{th}}$  player in a Bayesian game  $(N, A, \Theta, p, u)$ , where the type of the  $i^{\text{th}}$  player is  $\theta_i$  and where the players' strategies are given by the mixed-strategy profile  $s$ , is defined as*

$$EU_i(s, \theta_i) = \sum_{\theta_{-i} \in \Theta_{-i}} p(\theta_{-i} | \theta_i) \sum_{a \in A} \left( \prod_{j \in N} s_j(a_j | \theta_j) \right) u_i(a, \theta_{-i}, \theta_i).$$

To sum up, the  $i^{\text{th}}$  player has to consider each possible assignment of types to all other agents and every pure action profile  $a$  to evaluate his utility in the *ex interim* setting.

Having defined a player's utility, it is now possible to define best response in a Bayesian game.

### 3. Background and Related Work

**Definition 3.1.10.** *The set of player  $i$ 's best responses to mixed-strategy profile  $s_{-i}$  are given by*

$$BR_i(s_{-i}) = \operatorname{argmax}_{s'_i \in S_i} EU_i(s'_i, s_{-i}, \theta_i).$$

Now it is possible to define Bayes-Nash equilibrium.

**Definition 3.1.11.** *A Bayes-Nash equilibrium is a mixed-strategy profile  $s$  that satisfies  $\forall i s_i \in BR_i(s_{-i})$ .*

#### Stackelberg Equilibrium

In security domain, game theoretic solution concept of Stackelberg Equilibrium has been very popular recently [41]. The concept assumes that in strategic interaction between two players one particular player is dominant. The dominant player computes its strategy and discloses it to the other player before the game starts. Then the other player, who is often called a follower, plays optimally with respect to the strategy of the dominant player [36]. In security applications, the concept of Stackelberg Equilibrium better describes the situation, when the attacker is able to discover the defender's strategy before performing the attack. The following definition is adapted from [66].

**Definition 3.1.12.** *Without loss of generality, let a leader be denoted as a player 1 and let a follower be denoted as a player 2. Let  $S_1$  be a set of possible strategies for the leader. Let  $b_2(s_1)$  denote the follower's best response to the leader's strategy  $s_1 \in S_1$ .*

*A strategy profile  $s = (s_1, s_2)$  is a Stackelberg Equilibrium (SE) if  $u_1(s_1, b_2(s_1)) \geq u_1(s'_1, b_2(s'_1)), \forall s'_1 \in S_1$ .*

In addition, if the follower breaks ties in favor of the leader, then the solution concept is called a *Strong Stackelberg Equilibrium* (SSE). Note that the effect of breaking ties in favor of the leader is not restrictive in practice: by using negligible perturbations, the defender can make a particular follower's pure action the only optimal one for the follower [36, 34].

## 3.2. Receiver Operating Characteristic

Receiver Operating Characteristic (ROC) is a technique for visualizing and evaluating machine learning classifier performance. The technique is primarily used for evaluation of binary classifiers and has particular advantages for domains with skewed class distributions, unequal classification error costs for different classes, and varying proportion of the classes [67]. The class distribution in case of O2 CZ fraud detection case is very much skewed<sup>1</sup>, costs for misclassifying a fraudulent instance and a legitimate instance differ, and in domain of fraud detection it is known, that proportion of fraud varies from month to month [11]. More importantly, the ROC curve enables to choose a setting of a classifier with acceptable false positive rate, which is a crucial requirement in practice [5, 35, 24, 13, 39, 29, 47, 28]. For these reasons the ROC curve was used for evaluation and choosing a machine learning model for the O2 CZ fraud detection module. Moreover, in this thesis we extend a state-of-the-art game-theoretic adversary-aware method for adversarial classification which is based on the ROC curve. We provide a brief overview of the ROC technique which is used in the adversarial classification method developed in this work.

---

<sup>1</sup>Approximately 9.5% of all instances correspond to fraudulent cases, see Section 2.2.1.

|                                     |                  |                  |
|-------------------------------------|------------------|------------------|
| true<br>label<br>predicted<br>label | <b>P</b> ositive | <b>N</b> egative |
| <b>P</b> ositive                    | <b>TP</b> count  | <b>FP</b> count  |
| <b>N</b> egative                    | <b>FN</b> count  | <b>TN</b> count  |

$$FPR = \frac{FP}{FP + TN}$$

$$TPR = \frac{TP}{TP + FN}$$

$$precision = \frac{TP}{TP + FP}$$

$$recall = sensitivity = TPR$$

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

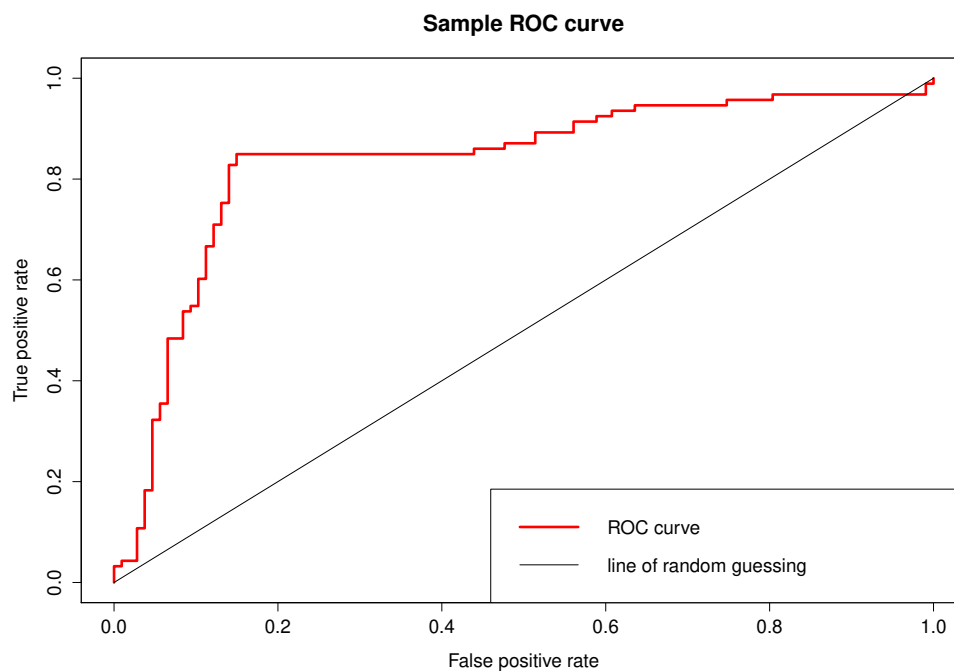
$$specificity = \frac{TN}{FP + TN} = 1 - FPR$$

Fig. 3.2.1. Contingency table and common performance measures calculated from it

### 3.2.1. Classifier Performance Measures

The classifier is a mathematical model used for mapping each instance to a predicted class label: either positive or negative. In Section 2.2.1 it was discussed that several state-of-the-art classification models naturally output a continuous quantity, a probability value or a score which represents the degree to which an instance is a member of a particular class, and final classification is made by thresholding the produced quantity [8, 13, 10, 14, 15]. Examples of the classifiers which output continuous scoring quantity might be: Naive Bayes classifier, neural networks and their ensembles, ensembles of classification and regression decision trees, such as random forests and boosted trees. Implementations of some machine learning models outputting discrete labels directly provide an option of predicting the class probabilities, e.g., see libsvm-based R package for Support Vector Machines [63]. In the following discussion we would, thus, consider a general classifier which outputs a scoring quantity describing the degree to which a classified instance belong to a particular class.

Given an instance to be classified, the classifier scoring output and the selected threshold on the output, there are four possible outcomes. If both the predicted class and the true instance label are positive, then it is counted as a *true positive* (TP). If the true instance label is positive, but the predicted class is negative, then it is counted as a *false negative* (FN). A *true negative* (TN) corresponds to a case when both the predicted label and the true label are negative. And the last case, when the true label is negative, however, the predicted label is positive, is termed a *false positive* (FP). Note that FP is commonly called a false alarm. Counts of TP, TN, FP, FN can be summarized in a contingency table, also called a confusion matrix [67]. Fig. 3.2.1 illustrates a contingency table together with formulas for commonly used performance measures, which can be calculated from the table. For the sake of conciseness, in the formulas total TP, TN, FP, and FN counts will be denoted as TP, TN, FP, FN correspondingly. FPR in the formulas stands for *false positive rate*. In literature FPR is sometimes called *false alarm rate*. TPR stands for *true positive rate*, also called *hit rate*, *recall*, or *sensitivity*. Sometimes in practice TPR is confused with precision, and precision in its turn might be confused with accuracy. For the sake of clarity, computation of all the measures is summarized in one place in Fig. 3.2.1.



**Fig. 3.2.2.** Sample ROC curve

### 3.2.2. ROC curve

In the previous paragraph we defined TPR and FPR for one particular contingency table corresponding to a specific classification threshold. However, for different classification thresholds, contingency tables would differ.

Without loss of generality, let us assume that the higher the scoring output of the classifier is, the higher the degree to which an instance belong to a positive class is. Therefore, it holds that the lower the classification threshold we choose, the higher TPR we obtain. This is something we desire. Unfortunately, this desirable result comes at a cost: the lower the classification threshold is, the higher FPR we obtain. High FPR is intolerable in practice and limiting FPR was argued to be a top priority for researchers in order to make adversarial classifiers widely deployable in real-world applications [5]. An ROC curve serves to illustrate a trade-off between costly FPR and desirable TPR in general. The ROC curve is a two-dimensional graph where TPR is plotted against FPR for different values of FPR, as a result of different thresholds. An example ROC curve is depicted in Fig. 3.2.2. The sample ROC curve was generated using ROCR R package and sample data provided with the package [68].

An idealized ROC curve depicts TPR for every possible value of FPR, having an infinite number of points. However, in practice an ROC curve is generated from a finite test set. As a result, a real-world ROC curve is a step function, which approaches a true ROC curve as a size of test set approaches infinity [67].

There are several important points on the ROC curve. The point (0,0) corresponds to the deterministic strategy of predicting negative label all the time, while the point (1,1) corresponds to predicting positive label all the time. The diagonal line  $TPR = FPR$  corresponds to the strategy of random guessing. For instance, a point (0.5, 0.5) corresponds to the strategy of guessing a positive label with probability 50%. Another point on the diagonal, a point (0.7, 0.7), corresponds to the strategy of guessing a pos-

itive label with probability 70%: if for each positive instance we choose a positive label with probability 70%, the expected TPR is 0.7, the same holds for negative instances and FPR. Having discussed the diagonal of random guessing, it is obvious that any classifier which corresponds to a point in the lower right triangle of the ROC space can be outperformed by random guessing. Finally, the point (0, 1) corresponds to perfect classification. An ROC curve of a perfect classifier would have  $\text{TPR} = 1$  for any FPR.

Note that the deterministic strategy of predicting a negative label all the time would gain accuracy of 91.5% on the skewed O2 CZ fraud data, however, an ROC curve of such strategy would consist of a single point (0, 0). For imbalanced datasets the ROC curve depicts performance of a classifier better than accuracy. However, the ROC curve is a two-dimensional graph, and it might be desirable to compare general performance of different classifiers based on a scalar value derived from the ROC curve. A common approach is to calculate the area under the ROC curve (AUC). The AUC has important statistical interpretation: probability that a randomly chosen positive instance would be correctly classified by a classifier is equal to the AUC of the classifier [67].

### 3.3. Related Work on Adversary-aware Classification Models

In this section we systematically review related work on game-theoretic adversary-aware machine learning models for detection of malicious behavior. The reviewed models are grouped based on the type of the used game-theoretic solution concept. When reviewing each related work model, we investigate properties of the model related to the goals of this work. It is shown that currently there is no model of adversarial classification which satisfies all the practical requirements formulated as goals of this thesis, see Section 1.2. Next, we categorize all the models based on the practical requirements on adversarial classification. Strengths and limitations of the models are summarized in one table, providing an overall overview to the reader. Finally, we analyze which models address some of the specified goals of the thesis. As a result of the analysis, we identify approaches from the existing state-of-the-art models we will capitalize our work on.

Before diving into models using both machine learning and game theory for adversarial classification, we first mention a notable study [22] of attacks against machine learning by rational adversaries. Even though the work does not use game-theoretic formalism to devise robust detection of malicious behavior, it formally models a rational adversary, defining his intention to avoid detection in terms of utility, and provides an algorithm for attacks by rational adversary against machine learning classifiers with differentiable discriminant functions. Finally, using the devised algorithm, in [22] it is shown that machine learning security systems can be easily evaded by attacks of the rational adversary. It is worth noting that while the work [22] focuses on attacks against machine learning system not using game-theoretic modeling of the interaction between the adversary and the system, there are game-theoretic studies which analyze general properties of the simplified attacker-system interaction without consideration of realistic details and constraints on classification methods [69, 38, 70].

#### Models of Problems Other than Exploratory Integrity Attacks

Now we start reviewing models which optimize detection of malicious behavior using game theoretic framework. As the framework is very broad, its usability is not limited just to optimization of adversarial classification by modeling strategic interaction between a security system and an adversary. For instance, in [39] cooperative game theory,

### 3. Background and Related Work

which is not within the scope of this work, is used to optimize intrusion detection using a network of sensors as a means of protection for a distributed system consisting of a set of subsystems. In [71] cooperative game theory is used to optimize configuration of an intrusion detection system which uses several libraries.

Security in the multiple-system setting can be improved by means of non-cooperative game theory as well. In recent works [47, 34, 37, 54] game theory was used to optimize protection of multiple users/systems against personalized spear-phishing attacks. The real-world requirement on low FPR was reflected in the defender’s cost in the models, without explicit control over FPR. The attacker’s utility was modeled realistically, assuming that the attacker prospers from false negatives of the system. However, the attacker’s intention to increase false negatives of deployed classifiers was not modeled in [47, 34, 37, 54]. The aim of this research was not to address exploratory integrity attacks, but to deal with a different problem: customization of user-specific security settings when the attacker strategically chooses a subset of users to attack.

#### NE-based Models

First, we review models using NE as a solution concept.

The problem studied in [46] is closely related to exploratory integrity attacks. However, instead of distinguishing malicious instances from legitimate ones, in the work [46] a classifier is designed to differentiate two types of malicious instances: spies vs. spammers. The adversarial classification setting is in principle the same as in the case of standard exploratory integrity attacks: the goal is to classify positive instances (spies) apart from negative instances (spammers), while positive instances intend to look like negative ones. The model assumes a linear classifier based on a single attribute: intensity of the attack. The attacker chooses the intensity of the attack and the defender in his turn chooses a classification threshold. False positives are penalized in the defender’s utility function, however, maximal expected FPR cannot be controlled. Strategic interaction is formalized as a normal-form game. NE of the model can be computed in polynomial time using LP solvers. The complexity might be reasonable for smaller instances of the problem. The resulting classification is randomized.

In [40] a standard problem of exploratory integrity attacks on supervised machine learning classifiers is addressed. Intentions and interaction between a rational adversary and a defender are modeled as a normal-form game. The machine learning method is restricted to linear classifier and the defender is constrained to choose deterministic classification. The adversary in the model is assumed to transform input data, and the defender chooses the classifier parameters. The adversary-aware model puts considerable restrictions on both utility functions and actions of both players: the joint action space is assumed to be compact and convex, utility functions of both players are assumed to be convex, twice differentiable and monotonic in the player’s actions. Under the idealized restrictions an algorithm for computation of NE is devised.

In [24] the problem of exploratory integrity attacks in adversarial classification was also addressed by modeling the strategic interaction using game theory. The attacker’s and the defender’s incentives were modeled by a utility-based approach. In the model the attacker prospers from false negatives and has to pay some cost for modifications of malicious instances, while the defender prospers from correctly classified instances and is assumed to pay for measuring features of the instances. The attacker’s action is to modify malicious instances, and the defender’s action is to choose a classifier. In [24] the existence of NE equilibrium of the defined game was proven. However, its calculation was not addressed due to computational complexity: even if all attributes of training



data are from finite discrete domains, number of actions is doubly exponential in the number of attributes. Leaving the game-theoretic adversary-aware model unsolved, the authors of [24] develop a computational procedure to address the following scenario: a defender deals with training data without taking an adversary into consideration, next the adversary avoids the deployed classifier as if the defender was unaware of his presence, finally the defender derives a new classification rule knowing that the data were altered by the adversary. The scenario is devised for a naive Bayes classifier. The work [24] focuses on the described single-shot scenario when the adversary avoids the deployed classifier just once. The repeated version of the scenario is briefly discussed as well.

#### SE-based Models

Various real-world security problems were formalized as Stackelberg games. For instance, Los Angeles International Airport Police scheduling problem or scheduling for the Federal Air Marshals Service were formulated as Stackelberg games [42]. SE is considered to be a more realistic solution concept for several adversarial classification settings as well, because an attacker might be able to determine the deployed security before performing an attack.

For that reason the authors of a NE-based model from [40] also modeled the interaction between the adversary and the defender as a Stackelberg game in [72]. In the Stackelberg model from [72] the defender is a leader. He chooses a classifier and the attacker plays his best response to the chosen classifier. Analogously to the NE-based model from [40], the model from [72] also restricts machine learning method to a linear classifier with a deterministic classification strategy. The defender's utility function is restricted to be convex and differentiable in the defender's action. Under the restrictions, authors devise an algorithm for SE computation.

In contrast with the reviewed Stackelberg model, in some studies the attacker was modeled to be a leader, who modifies data and discloses how the future distribution would differ from original data. In [72] this setting was argued to contradict the intuition of the adversarial classification problem: in the setting the defender just needs to solve a straightforward optimization problem, minimizing the risk on the transformed data under real-world restrictions, e.g. restriction on false-positive rate, see Fig. 2.3.1. For the purpose of completeness, the models with the counterintuitive formulation of a Stackelberg game are to be reviewed as well.

In [55] adversarial classification was modeled as a Stackelberg game, where an attacker was assumed to be the leader, who modifies data knowing that the defender would play the best response later. In practice this setting would mean that the defender would simply produce an optimal classifier based on the new observed data and possible practical constraints. In the model classification method is restricted to a threshold-based linear classifier with data having a single continuous feature. Interaction between players is modeled by an extensive-form game with perfect information, where each player performs a single action. Even in this toy setting there is an infinite amount of possible transformations by the attacker, making the problem computationally intractable. For this reason a subgame-perfect SE in [55] is estimated using a genetic algorithm.

A conceptually similar model of adversarial classification was developed in [73]. Again, the attacker is assumed to be the leader, the machine learning method is restricted to a linear classifier, training data are assumed to be one-dimensional, the

### 3. Background and Related Work

attacker’s action is to modify the data, the defender’s action is to train an optimal classifier on the modified data, and SE of the game is estimated with evolutionary computation.

A Stackelberg game with the attacker moving first was also presented in [21]. The game is modeled as an extensive-form game with a single move from both sides. The model restricts machine learning method to a linear classifier and requires small number of features in training data.

Lastly, adversarial classification was modeled as an analogous extensive-form Stackelberg game with a rational attacker being a leader also in [25]. The defender’s utility was derived from a total number of misclassified instances, while the attacker’s utility was based on a number of false negatives. Due to an infinite number of possible attacker’s transformations of the data, the model was shown to be analytically intractable even in case of a linear classifier. A subgame-perfect SE of the game was estimated using a stochastic search algorithm.

Having reviewed models with counterintuitive formulation of a Stackelberg game, we now proceed with overview of the most advanced adversary-aware models of machine learning classification in security domains.

Authors of a recent work [36] use a normal-form game model to formalize the strategic interaction between an attacker and a defender during exploratory integrity attacks. The model enables to derive randomization of a classifier decision boundary. A notable property of the model is that it allows usage of any machine learning approach for classification of malicious behavior, without any restrictions on training data either. The defender’s overall cost is realistically derived from operational costs due to FPR and FNR, and from costs of expected attacks by a rational adversary. In the model the defender’s costs due to FPR and FNR are estimated directly from an ROC curve of the used machine learning classifier. In [36] it is assumed that all malicious instances from training data are stationary, however, there is a rational attacker which has an incentive to be undetected. The rational adversary is modeled to prefer cases when the nondetection takes place for a classifier setting corresponding to high FPR. Note that the rational adversary in the model can be viewed as an adaptive malicious entity in terms of Section 2. Note that in [36] both SE and NE of the model were studied. It was shown that a fine discretization of ROC suffices to derive randomized classification close to the optimal one. As it was discussed in Section 3.2, every real-world ROC curve is discrete and it approaches an ideal ROC curve as the amount of test data used to produce the ROC curve approaches infinity. Therefore, with sufficiently large test datasets the approach from [36] enables computation of near-optimal randomization of the classifier decision boundary. Computation of SE was based on LP solver, and computation of NE was addressed using a standard game theoretic solver. In other words, the used NE computation procedure belongs to PPAD complexity class [74].

In another recent work [35] the problem of exploratory integrity attacks on supervised machine learning classifier was also formulated as a Stackelberg game. In [50] computational tractability of the model was improved. As in [36], the formulated game-theoretic model from [35] provides remarkable flexibility: any machine learning method can be used as a black box. Moreover, modeling of the rational attacker’s utility is quite advanced, taking into consideration variability of different malicious rational entities and their current feature vectors. It is realistically assumed that the goal of each rational entity is to avoid detection by means of the smallest modification of the current feature vector. The defender’s utility is based on positive gains from TNs and losses due to

FNs. The model enables to set an expected probability that a random malicious entity is adaptable. SE of the model is computed using LP solver. However, computational tractability of the formulated LP is the main issue of the model. In the model randomized classification decision is optimized personally for each possible input vector. In other words, in the initial model there is a variable for every possible feature vector of an instance to be classified. This impressive modeling flexibility is a double-edge sword, making the initial LP infinite in case of continuous features and making it intractably large even in case of binary features. In [50] LP formulation and its computational tractability was improved for the case of a binary features vector, making the model applicable in use cases where restriction on all attributes to be binary is reasonable. For instance, efficacy of the optimized approach was demonstrated on a spam filter data with each email message encoded using the bag-of-words model [50]. Moreover, to the best author's knowledge, the model from [35, 50] is the first adversary-aware classification model directly incorporating operational constraints in it. In the model a total number of alarms can be restricted. Even though the authors argue that in practical applications of adversarial machine learning it is FPR which needs to be restricted in the first place[50], the model does not enable limiting FPR directly. In real-world applications like the O2 CZ fraud detection it is required from the machine learning system to detect as many malicious instances as possible. However, in practice a high FPR results in the denial of service [5].

Reviewing two advanced models concludes the overview of related work on adversary-aware machine learning models.

#### 3.3.1. Categorization of The Models

In the previous section adversary-aware machine learning models for detection of malicious behavior were reviewed. All the models are to be categorized based on properties required in real-world applications of adversarial classification. The properties of the models are summarized in Table 3.3.1. The model categorization structurally summarized the presented overview and is based on applicability of the model to the problem of exploratory integrity attacks on machine learning systems in real-world settings analogous to the O2 CZ fraud detection case. Thus, if some model was shown to be intractable even using the simplest machine learning method on a restricted training data, then from the perspective of practical applicability we categorize the model to be restrictive regarding the machine learning method and training data even though it was not directly incorporated in the formal model.

#### 3.3.2. State-of-the-art to Capitalize the Novel Model on

In practice it is required to use the most accurate state-of-the-art machine learning approaches, in order to derive the best possible predictive model. Only two recent models due to [36] and [35, 50] enable to derive a robust adversary-aware classification based on any state-of-the-art machine learning methods, see the summarizing Table 3.3.1. The model from [35, 50] provides remarkable modeling power, however it is inapplicable if some features of training data are continuous, which makes it impossible to use the model for the O2 CZ fraud detection case without hurting the machine learning predictive model<sup>2</sup>. Another notable model from [36] does not have this limitation. On

---

<sup>2</sup>Discretization of all continuous attributes and establishing a binary feature for each possible value of each attribute results in significant and unnecessary decrease of the machine learning classifier performance due to the curse of dimensionality and information losses during the discretization.

### 3. Background and Related Work

| Models From Work       | Modeling exploratory integrity attacks (detection avoidance) | No restrictions on machine learning method     | No restrictions on training data | Computational efficiency            | Control over FPR               | Control over expected adversarial rationality | Randomized classification | Variability of malicious rational entities modeled |
|------------------------|--|--|----------------------------------|-------------------------------------|--------------------------------|---|---------------------------|--|
| [39]                   | <i>different problem: distributed system</i>                 |  |                                  |                                     |                                |   |                           |  |
| [71]                   | <i>different problem: combining libraries</i>                |  |                                  |                                     |                                |   |                           |  |
| [47], [34], [37], [54] | <i>different problem: personalized security</i>              |  |                                  |                                     |                                |   |                           |  |
| [46]                   | ✓  |  |                                  | ±                                   |                                |   | ✓                         |  |
| [40]                   | ✓  |  | ✓                                | ✓(under the idealized restrictions) |                                |   |                           |  |
| [24]                   | ✓  | <i>equilibrium of the problem not resolved</i> |                                  |                                     |                                |   |                           |  |
| [72]                   | ✓  |  | ✓                                | ✓(under the idealized restrictions) |                                |   |                           |  |
| [55]                   |  |  |                                  |                                     |                                |   |                           |  |
| [73]                   |  |  |                                  |                                     |                                |   |                           |  |
| [21]                   |  |  |                                  | ?                                   |                                |   |                           |  |
| [25]                   |  |  | ✓                                |                                     |                                |   |                           |  |
| [36]                   | ✓  | ✓  | ✓                                | ±                                   |                                |   | ✓                         |  |
| [35, 50]               | ✓  | ✓  |                                  | ± (with binary features only)       | control over total alarms only | ✓   | ✓                         | ✓  |

**Tab. 3.3.1.** Models using both machine learning and game theory for adversarial classification

the other hand the model from [36] does not address variability of various malicious rational entities<sup>3</sup>, neither it enables to set a probability that a malicious entity is adaptive. Therefore, advantages of both models should be combined in order to improve the state-of-the-art adversary-aware classification. Moreover, in order to derive not only accurate, but also an applicable in practice method, it is required to incorporate control over expected FPR into a novel model.

The model due to [36] is the only one directly applicable on training data for the O2 CZ fraud detection with usage of arbitrary state-of-the-art machine learning classifier. As a result, this model would be taken as a main predecessor for a new model. The novel model will be based on the classifier’s ROC curve as well. Moreover, this would enable to naturally incorporate FPR into the defender’s utility function. The predecessor model is to be extended by including advanced modeling of various types of adversarial entities, data-driven estimation of adversarial utilities, and control over expected adaptability of rational entities, all these extensions are inspired by the model from [35, 50] mainly. Furthermore, the new model must be extended by incorporating user-specified control

<sup>3</sup>The model due to [36] assumes that all malicious rational entities are identical having the same incentives, while addressing the variability was left for the future work.

over the expected FPR in it. At the same time, in the original model computation of NE solution concept is done with a general algorithmic procedure belonging to PPAD complexity class and computation of the SE solution relies on solving a finite number of subproblems with standard LP solvers. Thus, when developing a novel model by analogy with the existing model, no computationally efficient approach for NE would be provided as a free lunch. Moreover, knowing that in special cases SE of security games can be computed very efficiently [43], it would be also necessary to analyze possible ways to improve SE computation in the model. The challenging task of discovering efficient algorithms for computation of the solution concepts is a subject of Chapter 5. The subject of the next Chapter 4 is to address in a single model all the discussed requirements.

### 3. *Background and Related Work*

## 4. Novel Model of Adversarial Classification

In this chapter we design and formulate a novel model which would be used to address the practical requirements on an adversarial classifier listed as goals in Section 1.2.1.

### Introduction

In order to build a robust protection against a real-world population of rational adversarial entities, a designer of security must collect information about the adversarial population and realistically model what the rational enemies desire, what they are able to do, what the designer himself prefers and is capable of.

To build a realistic model, it must be based on real-world observations. The model from [35] introduces realistic modeling of the diverse attacker population and interaction with it. In [35] it is noted that variability of rational entities in real-world adversarial classification is given by differences in personal properties and goals of adversarial entities, which take place in practice. The authors of the work [35] define a type of the attacker based on his feature vector. Inspired by the idea from [35] we develop a model using collected data about adversarial population to address the variability of adversarial entities. Data about a rational entity, collected when there was no machine-learning defense, reveals information about the particular type of a rational adversary showing his ideal method of attack.

In [35] a feature vector of a malicious entity is used to define the type of the adversarial entity. The model then aims to derive an optimal security for each possible type. To put it differently, the model from [35] seeks to derive a defender's decision for each possible input feature vector separately. For real-world applications with infinite feature spaces, for instance for the O2 CZ fraud detection case, the model leads to an infinite number of adversarial types making it impossible to compute the formulated security strategy<sup>1</sup>. Comparing a mathematical model of adversarial classification with a weapon, the sharpest sword is of no practical use if it is so extremely heavy that a warrior cannot raise it off the ground. Figuratively speaking, a handleable wooden fighting stick might be more useful in practice. A finite model with a tractable size from [36] can be used to model the O2 CZ fraud detection case. However, for now it lacks modeling of the adversarial variability and efficient algorithms for computation of the solution concepts. The goal of this chapter is to come up with a tractable finite model enriched with data-driven estimation of the variability of the attacker, as well as his intentions and capabilities.

The chapter is structured as follows. First, we formulate a Bayesian game model which can be considered an extension of the model from [36] with adversarial types from [35]. Next, to improve tractability of the solution computation in practical settings, we simplify the modeling while preserving the data-driven estimation of the

---

<sup>1</sup>Even though for real-world cases of finite feature vector spaces the model from [35] is also impractical, as it leads to an intractably large LP formulation, in [50] computational feasibility of the model was improved for applications with binary features.

adversary variability. The model is transformed to a normal-form game of a tractable size. Notably, the novel model exhibits the same mathematical properties as the model from [36]<sup>2</sup>. Therefore, the proposed model can be viewed as a special case of the general model from [36]. At the same time, in term of modeling power the new model can be considered an extension of the model from [36] to modeling in the style of Bayesian game with a newly introduced data-driven approach. Note that the most important notations introduced and described in this chapter are summarized in Appendix B for quick reference. To keep the work illustrative and assumptions realistic, the proposed data-driven utilities estimation is presented in the context of the O2 CZ fraud detection case.

### 4.1. Bayesian Game of Data-driven Security against Varying Adversary

#### 4.1.1. Players

We model adversarial classification as a game between two players: a defender and an attacker.

#### 4.1.2. Actions

As in [36] we assume that each player chooses a classification thresholds from a set of thresholds  $\mathbf{T}$ .

#### Defender

In [36] the game has continuous strategy spaces which are later discretized and the discretized version of the game is extensively analyzed. Both in the novel model and the model from [36] the defender's available actions are derived based on an ROC curve. Due to the fact discussed in Section 3.2 that each real-world ROC curve has a finite number of points, in contrast with the model [36] we deal with finite strategy spaces from the beginning. Let  $\mathbf{F}$  be a finite set of all FPR's of points specifying the real-world ROC curve<sup>3</sup>. And let  $\tau$  denote a mapping from the set of FPR's onto a set of classification thresholds. We define  $\mathbf{T} = \tau(\mathbf{F})$ . The defender's action is thus to choose a classification threshold  $t_d \in \mathbf{T}$  and classify all instances using the chosen threshold. Note that in [35] the defender chooses a different classification rule for each possible instance, or in other words for each possible type of the attacker<sup>4</sup>. As a result, such approach leads to an infinite model in case at least one feature in the dataset is continuous. We rather choose to base our model on the observed instances and estimate properties of the population of adversaries using collected data. Thus, we define the types of the attacker based on the observed malicious instances only. Note that this way we avoid hypotheses unfounded with observations. At the same time if in such data-driven modeling the defender's action is to choose a classification rule for each known attacker type, it restricts the defender to the already observed malicious feature vectors only, and as a result the model would likely become of limited practical use in real-world

<sup>2</sup>In the following chapter it will be shown how the newly derived utilities can be mapped to utilities from [36].

<sup>3</sup>To be more specific, once a real-world ROC curve is generated,  $\mathbf{F}$  can be obtained as follows: for each unique TPR value of the generated step function, consider the minimal FPR.

<sup>4</sup>Remember, in [35] attacker types are given by unique feature vectors.



settings as the classifier would assume to be legitimate each feature vector not observed as malicious previously. Therefore, our modeling choice of the defender’s pure strategy is to set a general classification rule based on the observed data describing malicious and legitimate population<sup>5</sup>. To sum up, the model is based on the collected dataset. Moreover, we assume that the observed data provide us some general information about the population of attackers and based on this information we derive a general security against the population, at the same time we do not assume that we have collected data on each possible attacker type<sup>6</sup>.

As it was discussed in more detail in Section 2.2.1, for the used classifier it must be possible to produce an ROC curve. In other words it must be possible to determine the degree to which a particular instance belongs to a malicious class. The defender classifies as malicious all instances with the danger degree higher than the chosen classification threshold. In the following text for the sake of conciseness we often call the degree to which a particular instance belongs to a malicious class the probability of being malicious.

### Attacker

Note that in the Bayesian model by an attacker we mean a particular malicious entity.

Analogously to [36], in the novel model an action of the attacker is to guess which classification threshold the defender has chosen and then to perform the exploratory integrity attack against the guessed threshold. In Section 2.3 it was discussed why in order to build the system secure from ground-up, the defender should not rely on the assumption that the attacker cannot know the used classification method. Based on the principle of security by design, it is assumed that the attacker is able to estimate the classifier output representing the degree to which an instance belongs to malicious samples. To put it in context, the adversary is assumed to be able to determine the probability that a particular email belongs to spam messages, or the probability that a customer would be considered fraudulent. To sum up, the pure strategy of the attacker is to guess thresholding on the probability predicted by the classifier and perform the avoidance attack against the chosen thresholding.

Lastly, it remains to mention what happens if the attacker guesses a threshold which can be never played by the defender. In [36] it was shown that a rational attacker would always guess thresholds which a defender might play, as guessing other thresholds are dominated actions for the adversary. Therefore, we assume that the attacker guesses the same thresholds as the defender. If the assumption does not hold, then the performance of the security system only improves.

To sum up, the attacker’s action is for each his type  $x$  to guess a classification threshold  $t_a(x) \in \mathbf{T}$  and perform detection avoidance. Let  $X$  denote a set of all attacker’s types. Then the attacker’s action is a mapping  $t_a : X \mapsto \mathbf{T}$ .

---

<sup>5</sup>Detailed modeling of the players’ utility functions is described in Section 4.1.4.

<sup>6</sup>An interesting extension of our model for the future work might be developing a data-driven approach based on clustering of the observed malicious instances and defining attacker types using the clusters. Perhaps, Gaussian Mixture Model (GGM) clustering might be a promising method of choice. Using the derived from collected data GGM it would be possible to estimate for each new instance its probability of being malicious and the cluster-based type of a probable attacker. Using this idea it might be possible to derive a type-specific security. It would be required to develop a method for estimation of the players’ utility functions from the suggested GGM-based attacker types.

### 4.1.3. Player Types

In [36] the model was motivated by security applications with a natural notion of the attack intensity. Examples of such applications can be brute-force password cracking or data exfiltration. The ideal attack for each rational entity in such applications correspond to the highest attack intensity and the intensity was assumed to be mapped onto the classification threshold. To put it differently, in terms of the model from [35] it was assumed in [36] that there is just one type of rational adversarial entities and this type corresponds to the highest intensity possible. We extend it to the setting when there are many types of the attacker. We define types of the attacker based on historical data about malicious population, inspired by the idea from [35]. It was discussed at the beginning of this section that an attacker feature vector observed before deployment of the classifier contains information about the optimal natural attack for a particular adversarial entity. Thus, each observed feature vector  $x$  defines a type of the attacker.

One of strong properties of the novel model is that it presents a data-driven approach to estimation of utilities. Let the dataset used for derivation of the model be called a validation dataset.

Thus, a set of all attacker's types  $X$  is a set of all malicious feature vectors in the validation dataset. All the entities are assumed to occur with the same probability, moreover, in real-world application with huge or infinite feature spaces it is very unlikely that in the collected data, feature vectors of malicious entities repeat. Therefore, if there are  $|X|$  malicious instances in the collected data, we assume that there are  $|X|$  different types of the attacker and each type occurs with probability  $p(x) = \frac{1}{|X|}$ .

Regarding the second player, there is just one type of the defender.

### 4.1.4. Utilities

#### Defender

In the case of the O2 CZ application<sup>7</sup> the defender's primary goal is to prevent losses due to fraud by detecting fraudulent entities. However, rational fraudulent adversaries tend to modify their feature vectors aiming to avoid detection [13, 28, 11]. In related work it was noted that for the attacker it is easier to avoid detection by changing his natural type for the type as close to the natural one as possible [40, 72, 21, 24, 35]. In the context of the O2 CZ case it holds that a customer very well estimates legitimate behavior patterns of people from his social group. However, if he aims to fake someone legitimate from a completely different social group, then it is a challenging task to precisely estimate the probability of the unknown type to be fraudulent. Consequently, the more extreme modifications of the feature vector the attacker performs, the less certain his success is. We use the same mathematical model for capturing uncertainty of severe feature vector modifications as in [35]. To be specific, let us assume that by the attack corresponding to a feature vector  $x'$  the attacker can steal amount of money  $G$  and that the attacker with a natural type corresponding to a feature vector  $x$  needs to modify his feature vector from  $x$  to  $x'$ , in order to avoid detection. To model the fact that the more extreme modification is required, the less certain is the attacker's success because it is harder for the attacker to reason about the unknown type of instances, we assume that even if the classification threshold is guessed correctly, in the described setting the attacker can steal  $G \cdot \exp^{-\gamma\|x-x'\|}$ , where  $\|\cdot\|$  stands for L2-norm and  $\gamma$  is a user-specified parameter [35]. The defender can loose exactly what the attacker can

---

<sup>7</sup>See Section 2.2.1 for the description of the O2 CZ use case.

steal. Thus, this modeling choice is reflected in the defender’s utility as well. This way we aim to avoid unrealistic overestimation of the attacker’s abilities, as discussed in Section 2.3.

As it was described in Section 2.2.1, the developed machine learning classifier among others uses two money-based features: the amount of money spent by the customer during the last 28 days  $u^{28d}$  and the amount of money spent during the last 24 hours  $u^{24h}$ . If a particular customer is not detected as malicious but in fact turns out to be fraudulent<sup>8</sup> it is assumed that the customer would be able to use the services during the month without intention to pay. The amount of money a fraudulent customer would be able to steal in the end of the month is estimated based on  $u^{28d}$  and  $u^{24h}$ . Money  $u^{28d}$  spent during the last four weeks by the customer who decided not to pay for the service part way through his contract might be a good approximation of the amount the customer would try to steal during the month. However, there is no historical information for customers who sign up for the service with no desire to pay. The amount of money such fraudulent customer would be able to steal during the next four weeks can be estimated as  $28 \cdot u^{24h}$ . To sum up, when a customer with a feature vector  $x$  is not detected as malicious but turns out to be a fraud with the natural type  $x'$ , then we assume the estimated gain of the fraud and therefore the estimated loss of the defender to be

$$l_d^{FN}(x) = \max\{u_x^{28d}, 28 \cdot u_x^{24h}\} \cdot \exp^{-\gamma \|x_n - x'_n\|}. \quad (4.1.1)$$

To address the fact that in reality not all adversaries are fully rational, we introduce into the model probability that an adversary is adaptive. Let it be denoted  $P_a$ . Note that  $P_a$  and  $\gamma$  both address real-world bounded rationality of adversarial population.  $P_a$  captures how likely an attacker of type  $x$  is adaptive, while  $\gamma$  captures another application-specific property: if an attacker of type  $x$  is adaptable, how much we expect him to modify the natural feature vector preserving the certain detection avoidance. Note that  $x_n$  denotes a feature vector  $x$  with all features normalized, to make modification of all features equivalent.

Also note that the estimation of a potential loss for a given fraud type  $x$  can be further improved. By collecting and analyzing corresponding data, it should be possible to build a predictive model which would in addition estimate, based on a specific type of a fraudulent entity  $x$ , how much money can be returned by the debt recovery. However, this is not within the scope of this thesis.

Next, if a fraudulent entity is correctly detected, then the loss is avoided,  $l_d^{TP}(x) = 0$ .

Neither the defender loses anything when a legitimate user is not detected as a fraud,  $l_d^{TN}(x) = 0$ .

Next, let us analyze the defender’s losses due to false alarms. In the model, analogously to existing related work [36, 35], we assume that the defender has to pay a constant cost for each FP,  $l^{FP} = c_{FP}$ . The cost can be derived based on internal expenses for the operation of the fraud detection department<sup>9</sup>. However, in reality wrong classification of a customer as fraudulent might result in losing the customer. It might be possible to collect relevant data and develop a model predicting a customer churn for a particular type  $x$  if wrongly declared a fraud. Yet, it is not within the scope of this thesis.

<sup>8</sup>See Section 2.2.1 for more information about the addressed type of telecommunications fraud.

<sup>9</sup>Note that there is an analogous operational cost for detecting TP. Yet, it is negligible compared to the money saved thanks to the detection. However, it is required to introduce  $c_{FP}$  into the model to capture the fact that FP comes with some cost.

#### 4. Novel Model of Adversarial Classification

Having discussed the defender's losses due to FP, FN, TP and TN, we now proceed to the total defender's costs when choosing a particular classification threshold. Let  $X_s^{FN}(t_d)$  denote a set of attacker types which would not be detected for the defender's thresholding  $t_d$  even if the rational attacker is static, not doing anything. The subscript  $s$  stands for *static*. It was discussed that each attacker's type has probability  $\forall x, p(x) = \frac{1}{|X|}$ . Thus, taking into account the attacker types probability, the defender's loss due to the undetected static attacker is  $\frac{1}{|X|} \sum_{x \in X_s^{FN}(t_d)} l_d^{FN}(x)$ .

Let  $X_a^{FN}(t_d, t_a())$  denote a set of the attacker types which would avoid detection due to adaptive action of the attacker. Remember that a classifier detects all instances with probability of being malicious exceeding the classification threshold. To put it differently,  $t_d < t_a(x) \implies x \notin X_a^{FN}(t_d, t_a())$ . Lastly, let  $X^{FP}(t_d)$  denote a set of FP's for the classification threshold  $t_d$ .

The defender's expected utility for choosing a classification threshold  $t_d$  when the attacker's action is  $t_a()$ , denoting a mapping  $t_a : X \mapsto \mathbf{T}$ , can be summarized as

$$U_d(t_d, t_a()) = -\frac{1}{|X|} \sum_{x \in X_s^{FN}(t_d)} l_d^{FN}(x) - |X^{FP}(t_d)| \cdot c_{FP} - \frac{P_a}{|X|} \sum_{x \in X_a^{FN}(t_d, t_a())} l_d^{FN}(x). \quad (4.1.2)$$

#### Attacker

The money which the defender loses due to adaptivity of the adversary, the rational attacker gains<sup>10</sup>. Therefore, the attacker's expected utility for his action function  $t_a()$  when a defender plays  $t_d$  is modeled as

$$U_a(t_d, t_a()) = \frac{P_a}{|X|} \sum_{x \in X_a^{FN}(t_d, t_a())} l_d^{FN}(x). \quad (4.1.3)$$

It is assumed that the higher the classification threshold the attacker successfully bypasses, the higher the reward for undetected attack he gains. It was first motivated in [36] by security applications with a natural dimension of the attack intensity [36]. The assumption can be also motivated with the intuition, that the higher the probability of being malicious the unstopped fraud had, the more damaging behavior in terms of financial losses it established. Another possible intuition is that we assume the classifier to distinguish the attackers from legitimate user, in other words we assume that for the attacker the most natural and desirable way of behaving corresponds to behavior patterns which a classifier detects as the attacker's ones.

#### 4.1.5. Bayes-Nash Equilibrium

It is possible to transform a Bayesian game into a normal-form game in such a way that NE of the resulting normal-form game is precisely Bayes-Nash equilibrium of the initial Bayesian game [65]. The core part of the transformation is done as follows: for each player  $i$  a set of his actions in the normal-form game is a set of distinct

<sup>10</sup>Note that the defender would lose some money even if the attacker is stationary. Analogously to [36] the stationary costs, not arising due to the rationality of the attacker, are not considered in the attacker's utility function, which is intended to capture consequences of actions by the rational attacker and his preferences over those.

mappings from the set of the player types to the set of his actions. For illustration purposes, consider a player with two types and three possible actions:  $A, B, C$ . Then in the induced normal-form game the player would have nine possible actions:  $AA, AB, AC, BA, BB, BC, CA, CB, CC$ . The action  $AC$  means that the player would choose an action  $A$  in case of his first type and would choose an action  $C$  in case of the second type.

As a result, a number of the attacker's actions in the normal-form game induced from the formulated Bayesian game would be  $|\mathbf{T}|^{|X|}$ , making this straightforward transformation intractable. At the same time, for real-world settings the formulated model might be solvable by transformation to extensive-form game and computing equilibria using the sequence form formulation. However, this is not within the scope of the thesis. We observe that for an attacker of any type the best response to an action by the defender is the same. Thus, by making an assumption that the attacker of any type always plays its best response to a particular defender's action should not result in significant losses of the simplified model performance. For this reason we proceed with the simplification and leave for the future work development of efficient algorithms solving the sequence form and addressing the restriction on FPR in it.

## 4.2. Normal-form Game

We are about to make a simplifying assumption in the developed Bayesian game model and formulate a normal-form game with the equivalent defender's actions, utilities and with analogous modeling of the rational attacker.

Observe, that if in the formulated Bayesian game the defender chooses a classification threshold  $t_d$ , then for the attacker the only dominant action is to guess the threshold  $t_d$  and modify all currently detected types  $x$  of the malicious instances. This follows from the modeling assumption that the higher is the probability of being malicious for the bypassed attacker, the higher the utility the attacker gains. Note that the defender detects all instances whose probability of being malicious exceed the threshold  $t_d$ . Motivated with the observation about the non-dominated attacker's action, we assume that the attacker guesses the same threshold no matter which type  $x$ .

Thus, after the simple modification the game transforms into a normal-form game. The structure of the normal-game remains very similar to the formulated Bayesian game: the set of players is the same and actions available to the players remain the same.

### 4.2.1. Utilities

Now, when the attacker guesses one threshold  $t_a$ , it is possible to simplify the defender's utility.

In the Bayesian game  $X_a^{FN}(t_d, t_a())$  denoted a set of adversarial types which for a classification threshold  $t_d$  would avoid detection thanks to attacker's adaptability. It held that  $t_d < t_a(x) \implies x \notin X_a^{FN}(t_d, t_a())$ . For the normal-form game the notation simplifies to  $X_a^{FN}(t_d, t_a)$ . Moreover, now it holds that  $t_d < t_a \implies X_a^{FN}(t_d, t_a) = \emptyset$ .

#### 4. Novel Model of Adversarial Classification

Therefore, the defender's utility function from Eq. 4.1.2 can be rewritten as follows:

$$U_d(t_d, t_a) = \begin{cases} -\frac{1}{|X|} \sum_{x \in X_s^{FN}(t_d)} l_d^{FN}(x) - |X^{FP}(t_d)| \cdot c_{FP} - \frac{P_a}{|X|} \sum_{x \in X_a^{FN}(t_d, t_a)} l_d^{FN}(x), & t_d \geq t_a; \\ -\frac{1}{|X|} \sum_{x \in X_s^{FN}(t_d)} l_d^{FN}(x) - |X^{FP}(t_d)| \cdot c_{FP}, & \text{otherwise.} \end{cases} \quad (4.2.1)$$

Analogously, the attacker's utility can be simplified:

$$U_a(t_d, t_a) = \begin{cases} \frac{P_a}{|X|} \sum_{x \in X_a^{FN}(t_d, t_a)} l_d^{FN}(x), & t_d \geq t_a; \\ 0, & \text{otherwise.} \end{cases} \quad (4.2.2)$$

#### On Usage of the Model in General

Note that in applications of adversarial classification without money-based attributes, when it is hard to correctly estimate losses due to different types of the attacker, it is possible to assume that each rational entity steals a single money unit if not detected. This way the model would capture the main attacker's desire to avoid detection.

## 5. Efficient Algorithms for Optimal Solutions

In this chapter we develop efficient algorithms for computation of optimal security strategies in the developed model based on both concepts of NE and SSE. Moreover, we address the restriction on the expected FPR.

The chapter is structured as follows. First, we show that the extended model has the same mathematical structure as the model from [36]. However, no model-specific efficient algorithm can be borrowed from [36] to determine solutions for the novel model, as computation of solutions in [36] was performed using general solvers. The main issue is that the used NE computation procedure belongs to PPAD complexity class [74]. Yet, neither a procedure for SSE computation optimized for the model has been developed yet. At the same time in order for the devised method to remain applicable in the future, efficient and scalable algorithms must be part of the approach. A current trend is to keep storing more and more data. As it was discussed in Section 3.2, with a size of a dataset approaching infinity, an ROC curve generated based on the data approaches a true continuous curve. Thus, we expect our algorithm to deal with significantly larger problem instances in the future, compared to the current state of affair.

Next, we develop a linear time algorithm for precise computation of the defender's strategy in NE. Furthermore, we address computation of SSE in the models. As a part of a general SSE computation for normal-form games, a set of LP's or a mixed integer linear program is usually solved. We analyze the model-specific LP formulation and develop linear time algorithm for solving the LP. This leads to a quadratic time algorithm for solving the SSE problem. The developed algorithms can be used to solve both the initial model from [36] and the novel model. Lastly, we develop an approximation algorithm to compute SSE randomized strategy satisfying the specified restriction on expected FPR, addressing the previously discussed practical requirement on adversarial classifiers to not produce a large amount of false alarms [5, 35, 24, 13, 39, 29, 47, 28]. The developed approximation algorithm addresses a problem in the style of the Neyman-Pearson approach to decision theory: the performance is optimized under the constraint on expected false positive rate.

### 5.1. General Model Structure

First, the mathematical structure of the initial model from [36] is presented. Next, it is shown that the novel model in fact provides a data-driven approach to estimation of utility functions from [36], having the same mathematical structure. We comment on a model notation which would be used in the following derivation of the algorithms.

#### Model from [36]

In the model from [36] the defender's overall costs for false positives and false negatives arising due to stationary population are denoted with  $c_d^b(t_d)$ , where  $t_d$  is a classification threshold chosen by the defender.  $c_d^b(t_d)$  is called the defender's background costs. In

[36]  $c_d^r(t_a)$  denotes the defender's non-decreasing costs for an undetected attack by a rational attacker, when the attacker guesses classification threshold  $t_a$ .

$r_a(t_a)$  denotes the attacker's non-decreasing and non-negative reward function for performing an undetected attack. In the case of a detected attack the attacker's non-negative penalty is assumed to be  $p_a$ . It is assumed that there is a single type of the attacker.

The sum of the defender's costs  $c_d^b(t_d)$  arising due to stationary population and costs  $c_d^r(t_a)$  due to the rational attacker gives the defender's total costs. In [36] the defender's utility is defined as

$$U_d(t_d, t_a) = \begin{cases} -c_d^b(t_d) - c_d^r(t_a), & \text{if } t_d \geq t_a; \\ -c_d^b(t_d), & \text{otherwise.} \end{cases} \quad (5.1.1)$$

The utility of the rational attacker is his reward for the undetected attacks and his penalty for detected ones:

$$U_a(t_d, t_a) = \begin{cases} r_a(t_a), & \text{if } t_d \geq t_a; \\ -p_a, & \text{otherwise.} \end{cases} \quad (5.1.2)$$

### Novel Model Rewritten

Comparing the presented model formulation from [36] and the novel model formulation from Eqs. 4.2.1, 4.2.2, we can see that the players' utilities in the novel model can be rewritten as the initial model Eqs. 5.1.1, 5.1.2, where the defender's background costs due to stationary population are

$$c_d^b(t_d) = \frac{1}{|X|} \sum_{x \in X_s^{FN}(t_d)} l_d^{FN}(x) + |X^{FP}(t_d)| \cdot c_{FP}, \quad (5.1.3)$$

for the defender's costs due to attacks by the rational adversary holds

$$t_d \geq t_a : c_d^r(t_a) = \frac{P_a}{|X|} \sum_{x \in X_a^{FN}(t_d, t_a)} l_d^{FN}(x), \quad (5.1.4)$$

the function of the attacker's reward for a successful attack

$$r_a(t_a) = \frac{P_a}{|X|} \sum_{x \in X_a^{FN}(t_d, t_a)} l_d^{FN}(x), \quad (5.1.5)$$

and finally the penalty for detection  $p_a = 0$ .

For the sake of compactness we will use in this chapter the notation  $c_d^b(t_d)$ ,  $c_d^r$ ,  $r_a$ ,  $p_a$ . The discussed meaning of the notation is summarized in Appendix B.

## 5.2. Nash equilibrium

In [36] a continuous set of thresholds was analyzed and it was proven that some dominated thresholds can never be a part of NE. We extend the analysis and prove that there is an additional group of thresholds which does not belong to NE support. Next, we continue analysis of the problem and develop a linear time procedure for NE computation.



### 5.2.1. Set of thresholds

Let  $U_d^c(t_d) = -c_d^b(t_d)$  denote the defender's utility for playing a threshold  $t_d$  in case an attack by the rational attacker is detected. In the case of an undetected attack the defender's utility is denoted by  $U_d^u(t_d, t_a) = -c_d^b(t_d) - c_d^r(t_a)$ . It is assumed that if the defender sets a fixed threshold, then the rational attacker would find out the threshold value and will successfully attack exactly on this threshold, causing the highest possible harm to the defender. Let  $U_d^u(t_d) = -c_d^b(t_d) - c_d^r(t_d)$  denote the defender's utility for playing a fixed threshold  $t_d$ .

For the defender it is possible to find an optimal non-randomised (i.e. fixed) threshold as:

$$t_d^* = \operatorname{argmin}\{c_d^b(t) + c_d^r(t)\}.$$

Hence,  $U_d^u(t_d^*)$  is a global maximum of  $U_d^u(t_d)$ . However, it might be profitable for the defender to play other thresholds instead of  $t_d^*$  in order to make the attack of the rational attacker detected. Yet, in [36] it was shown that in NE a rational defender would never consider to play any threshold  $t$  for which  $c_d^b(t) > c_d^b(t_d^*) + c_d^r(t_d^*)$ . Therefore,

$$U_d^c(t) - U_d^u(t_d^*) \geq 0 \quad (5.2.1)$$

is the defender's necessary condition for playing threshold  $t$  instead of playing  $t_d^*$ . In other words, a rational defender would never play a threshold  $t$  if the corresponding background costs  $c_d^b(t)$  is higher than the total costs ( $c_d^b(t_d^*) + c_d^r(t_d^*)$ ) for playing the optimal fixed threshold  $t_d^*$ .

Motivated by real-world situations, a bounded and closed set of all possible thresholds can be considered [36]. Let  $\mathbf{T}$  denote the set of all possible thresholds. Let  $\mathbf{T}_0 \subseteq \mathbf{T}$  be a set of the thresholds such that for  $\forall t \in \mathbf{T}_0 : U_d^c(t) - U_d^u(t_d^*) \geq 0$ . In other words  $\mathbf{T}_0$  denotes a set of thresholds considered for computation of NE strategies in [36]. We show that some thresholds from the set  $\mathbf{T}_0$  are dominated.

**Proposition 5.2.1.** *A defender cannot improve his utility by playing a threshold  $t_d$  instead of playing  $t'_d$  if  $t_d > t'_d \wedge c_d^b(t_d) \geq c_d^b(t'_d)$ .*

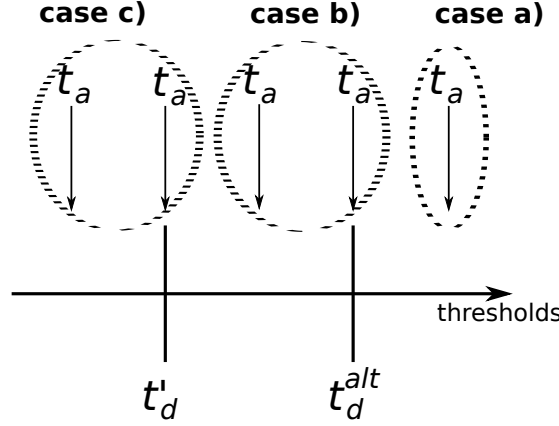
**Proof:** A set  $\mathbf{T}_0$  can be mapped to interval  $\langle 0, 1 \rangle$  [36]. Without loss of generality let us assume that all thresholds from the set  $\mathbf{T}_0$  form an interval  $\langle 0, 1 \rangle$ . If  $t'_d = 1$ , then the Proposition obviously holds. Suppose that  $0 < t'_d < 1$ . Let  $t_d^{alt}$  be such a threshold that  $t'_d < t_d^{alt} \wedge c_d^b(t_d^{alt}) \geq c_d^b(t'_d)$ . Several cases of attacks by a rational adversary are possible in relation to  $t'_d$  and  $t_d^{alt}$ , see Fig. 5.2.1.

Let  $t_a$  denote a threshold the rational adversary attacks on. The **case a)**, arises when  $t_d^{alt} < t_a < 1$ . For both defender's thresholds  $t'_d$  and  $t_d^{alt}$  the attack of the rational adversary is detected. Hence, the defender's costs due to rational attacker is zero. Due to background costs, a rational defender has no incentive to play the threshold  $t_d^{alt}$ .

In the **case b)**, when  $t'_d < t_a \leq t_d^{alt}$ , the defender's costs corresponding to the threshold  $t'_d$  are still  $c_d^b(t'_d)$  while the defender's costs corresponding to  $t_d^{alt}$  are  $(c_d^b(t_d^{alt}) + c_d^r(t_a))$ . As  $c_d^r(t)$  is a non-negative function and  $c_d^b(t_d^{alt}) \geq c_d^b(t'_d)$ ,  $(c_d^b(t_d^{alt}) + c_d^r(t_a)) \geq c_d^b(t'_d)$ . Again, a rational defender has no incentive to play the threshold  $t_d^{alt}$ .

In the **case c)** the attack is always undetected and the defender has pay costs  $c_d^r$  in both case. Further reasoning is the same, as in the **case a)**.

To sum up, the proposition 5.2.1 holds in all cases. ■



**Fig. 5.2.1.** Possible attacks by a rational adversary in relation to  $t'_d$  and  $t_d^{alt}$

**Corollary 5.2.1.** *If  $t_d^{max}$  is the smallest threshold corresponding to the global minimum of the defender's background costs  $c_d^b(t)$  on the set  $\mathbf{T}_0$ , then a rational defender cannot improve his utility by playing any threshold  $t$ ,  $t > t_d^{max}$ , instead of playing  $t_d^{max}$ .*

Note that on the bounded and closed set  $\mathbf{T}_0$  the function  $c_d^b(t_d)$  must attain a minimum at least once.

**Corollary 5.2.2.** *If the function  $c_d^b(t)$  of the defender's background costs is non-decreasing on set  $\mathbf{T}_0$ , then the defender's best response to any attacker's strategy is to play the lowest threshold  $t$  from set  $\mathbf{T}_0$ ,  $t = \min\{t \mid t \in \mathbf{T}_0\}$ .*

To sum up, a set of thresholds a rational defender should consider can be reduced.

As a result of Eq. 5.2.1, Proposition 5.2.1, Corollary 5.2.1, a restricted set of thresholds can be obtained. Let this subset of thresholds be called a set of reasonable thresholds  $\mathbf{T}_r$ . If  $c_d^b(t)$  has a graph depicted in Fig. 5.2.2, then the set of reasonable thresholds corresponds to solid-line intervals on the x-axis. The defender's dominated actions are not taken into consideration.

**Corollary 5.2.3.** *The defender's background costs  $c_d^b(t)$  is non-increasing on set  $\mathbf{T}_r$ .*

### Discretization

In [36] thresholds represent a continuous quantity. Capitalizing on this work, we have proven the discovered properties for the continuous case. Yet, the properties analogously hold for the case of a finite threshold set. Note that in [36] it was shown that discretization of a continuous interval of thresholds provide a reasonable approximation to the continuous problem formulation, i.e. there exists such a discretization that an error of approximation would be reasonably low, see [36] for details. At the same time due to the discussed in Chapter 4 fact that a real-world ROC curve is a step function, the realistic novel game model deals with the finite number of thresholds from the beginning. Hence, in the following a finite set of thresholds is considered, which can also be viewed as a discretization of the continuous interval of thresholds.

If the defender plays thresholds  $t_1$  and  $t_2$  with non-zero probability and at the same time he never plays thresholds from the interval  $(t_1, t_2)$ , then the rational attacker

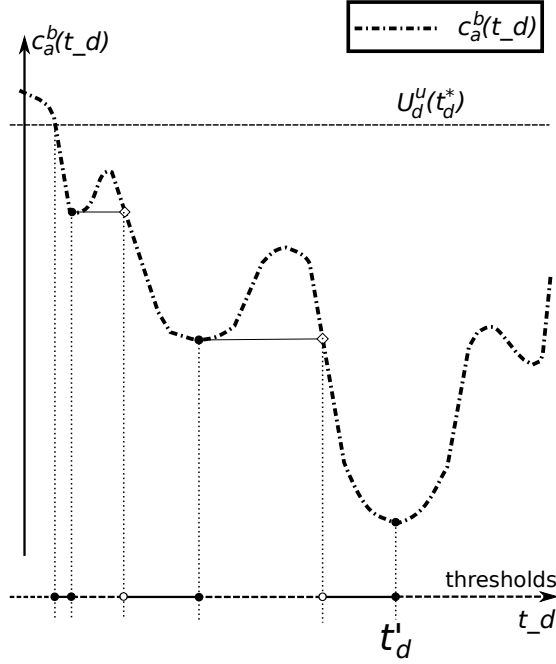


Fig. 5.2.2.  $c_a^b(t)$  and a corresponding set of reasonable thresholds

would consider playing only the bounds of the interval,  $t_1$  or  $t_2$ , in order to best respond [36]. It obviously follows from the monotonicity of the attacker's reward. By a discretization of a continuous interval of thresholds the defender can choose a set of thresholds which would be attacked by the rational adversary. Furthermore, due to the discovered threshold domination, it is sufficient to consider only thresholds from the set of reasonable thresholds in order to compute NE. Note that only dominated strategies were filtered out during formation of the set of reasonable thresholds  $\mathbf{T}_r$ , hence, at least one NE has its support from  $\mathbf{T}_r$ . To sum up, in the following text we focus on the set  $\mathbf{T}_r$ , in order to compute NE.

### 5.2.2. Computation of the players' strategies

An important fact about the defender's support of NE can be proven.

**Proposition 5.2.2.** *The support of the defender's strategy in NE always contains the highest threshold from the set of reasonable thresholds.*

**Proof:** The proposition can be proven by contradiction. Let us assume that in NE the defender plays the highest reasonable threshold with zero probability. Hence, the attack on the highest threshold is always detected. As a result, in NE the attacker never plays the highest threshold if it is not played by the defender. Hence, in NE neither the defender, nor the attacker plays the highest threshold with non-zero probability.

The considered defender's strategy cannot be the best response to the considered attacker's strategy. The defender can improve his expected utility by starting playing the highest reasonable threshold instead of the highest threshold which is currently played with non-zero probability. The defender cannot detect any attack while playing this threshold. Neither he can while playing the highest available threshold. However, in the latter case the defender's background costs becomes lower. To sum up, the defender's mixed strategy when the highest reasonable threshold is played with zero probability

cannot be the best response to the corresponding attacker's strategy. This contradicts the assumption that in NE the defender plays the highest reasonable threshold with zero probability. ■

Using the described finding, an algorithm for computation of the attacker's strategy in NE can be formulated.

The attacker has to make the defender indifferent on the support of NE. The defender's support of at least one NE is a subset of the set of reasonable thresholds. Moreover, the highest reasonable threshold has to be contained in the defender's NE support by Proposition 5.2.2. Hence, in NE the attacker has to make all reasonable thresholds at most equally appealing to the defender as the highest one.

Computation of such attacker's strategy can be done in an elegant and easy way. First, we construct initial attacker's strategy. The strategy support is tested whether or not it corresponds to a NE strategy. While it is not satisfied, the strategy is iteratively modified. Namely, the algorithm starts with the attacker's strategy when he plays only the highest reasonable threshold. If the strategy has to be modified, the attacker takes into consideration next reasonable threshold and makes all considered thresholds equally appealing to the defender. The only yet unconsidered threshold which might be more attractive for the defender compared to all considered ones would be the highest unconsidered threshold. The algorithm terminates due to non-increasing defender's background costs.

**Proposition 5.2.3.** *Let us number thresholds in descending order starting with an ordinal number 1<sup>1</sup>. The following Algorithm 1 will compute the attacker's strategy  $\sigma$ . Moreover, it will return an ordinal number  $N$  of the lowest threshold in the defender's support of NE.*

```

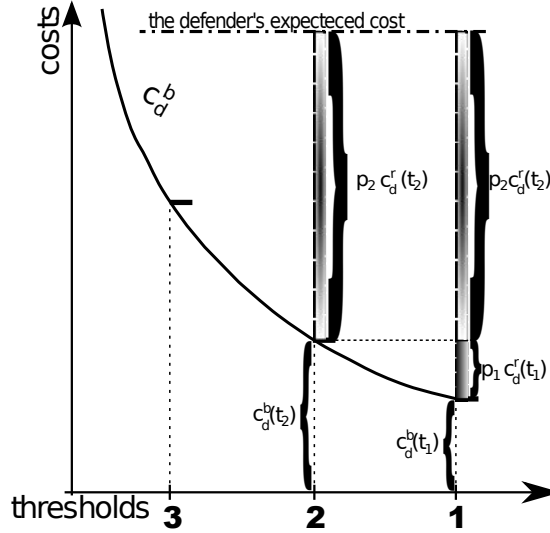
1  $N = 1$ ;
2  $\sigma_1 = [1]$ ;
   /* helping variable containing probability of playing the last added
   threshold */
3 last_probability = 1;
   /* for the sake of compactness assuming condition evaluation from
   left to right: if  $N =$  number of thresholds, then no threshold
    $t_{N+1}$  exists */
4 while  $N <$  number of thresholds and  $c_d^b(t_{N+1}) < c_d^b(t_N) + \sigma_N c_d^r(t_N)$  do
   | /* variable containing the probability required to equal the next
   | threshold */
5   next_probability =  $(c_d^b(t_{N+1}) - c_d^b(t_N)) / c_d^r(t_N)$ ;
6   last_probability -= next_probability;
7    $\sigma_N =$  next_probability;
8    $\sigma_{N+1} =$  last_probability;
9    $N = N + 1$ ;
10 end
11 return  $\sigma, N$ ;

```

**Algorithm 1:** The algorithm for determining the players' support of NE and the attacker's strategy.

---

<sup>1</sup>This way the first threshold corresponds to zero on the x-axis of the ROC curve.



**Fig. 5.2.3.** The defender's expected costs in the case of considering only two first thresholds.

**Proof:** First, it is checked if there are more thresholds than just one in the set of reasonable thresholds. In case there is no threshold more appealing to the defender than the highest one, the only reasonable threshold is played by both the defender and the attacker with probability 1.

If there are two or more thresholds in the reasonable set and the second threshold is more appealing to the defender than the first one, then in the first step the attacker has to compute probabilities  $p_1$  and  $p_2$  of playing the first and the second thresholds respectively. The probabilities  $p_1$  and  $p_2$  must be computed in such a way that both the first and the second thresholds would be equally appealing to the defender.  $(p_1, p_2)$  would be the next attacker's strategy estimation. Remember, that when the defender plays the first threshold, neither attacks on the first threshold nor attacks on the second threshold can be detected. On the other hand, if the defender plays the second threshold, then only attacks on the second threshold cannot be detected. Taking into account background costs, the defender's expected costs for playing the first threshold can be computed as follows:

$$c_1 = c_d^b(t_1) + p_1 \cdot c_d^r(t_1) + p_2 \cdot c_d^r(t_2). \quad (5.2.2)$$

Analogously, for the expected costs for playing the second threshold we have

$$c_2 = c_d^b(t_2) + p_2 \cdot c_d^r(t_2). \quad (5.2.3)$$

From the equation  $c_1 = c_2$  it can be easily obtained, that

$$p_1 = \frac{c_d^b(t_2) - c_d^b(t_1)}{c_d^r(t_1)}, \quad (5.2.4)$$

see Fig. 5.2.3.

One terminating condition for the algorithm is based on a total number of reasonable thresholds. If there are only two thresholds, then the attacker's strategy in NE is computed in this case.

However, when there is the third reasonable threshold the defender can play, then

## 5. Efficient Algorithms for Optimal Solutions

the current attacker's strategy cannot form NE if

$$c_d^b(t_2) + p_2 \cdot c_d^r(t_2) > c_d^b(t_3),$$

see Fig. 5.2.3. If the condition holds the defender would prefer to play the highest threshold with zero probability, which contradicts Proposition 5.2.2. However, if the condition does not hold, then the algorithm terminates.

This terminating condition remains the same when more thresholds have been already made equally appealing to the defender. It is so because the defender's expected utility of playing any of these thresholds, including the lowest one, is equal to the expected utility of playing the highest threshold.

Obvious yet important fact is, that once computed, the attacker's probability of playing a higher threshold remains the same in all next iterations of the algorithm. The defender's expected costs corresponding to all  $n$  thresholds have to be the same, therefore, the expected costs have to be equal pairwise. ■

The lowest and the highest thresholds from the defender's support of NE were obtained. They define an interval of thresholds the attacker makes the defender indifferent on. In order for the computed attacker's strategy to be NE strategy, the defender must in his turn make the attacker indifferent on the same interval of thresholds. Otherwise the attacker would prefer to play the action with the highest expected utility instead of the previously computed strategy.

Computation of the defender's strategy can be done in an elegant way as well. Basic intuition about the computation is to be provided. The attacker's reward function is non-decreasing on the set of reasonable thresholds. Furthermore, the attacker can always gain a reward for attacking on the lowest threshold, as such attacks cannot be detected. Therefore, the defender would make the attacker's expected utility of playing any higher threshold equal to the attacker's reward for playing the lowest threshold from the support. Knowing the lowest threshold from the support, in the first step the defender makes the attacker indifferent between two lowest thresholds. Let there be  $n$ ,  $n \in \mathbb{N}$ , thresholds in the players' support of NE. In order to make the  $n^{\text{th}}$  and the  $(n-1)^{\text{th}}$  thresholds equally appealing to the attacker, the defender has to find out how often the  $(n-1)^{\text{th}}$  must be left unprotected using the equation

$$r_a(t_n) = p_{n-1,2} \cdot r_a(t_{n-1}) - (1 - p_{n-1,2}) \cdot p_a, \quad (5.2.5)$$

where  $p_{n-1,2}$  is the defender's probability of playing the  $(n-1)^{\text{th}}$  threshold when just two lowest thresholds are considered. Note that from the attacker's point of view  $p_{n-1,2}$  is probability that the  $(n-1)^{\text{th}}$  threshold is unprotected. Also note that the right side of Eq. 5.2.5 represents the attacker's expected utility for playing the  $(n-1)^{\text{th}}$  threshold. From Eq.5.2.5

$$p_{n-1,2} = \frac{r_a(t_n) + p_a}{r_a(t_{n-1}) + p_a}. \quad (5.2.6)$$

And  $p_n = 1 - p_{n-1,2}$ . If the defender makes the attacker indifferent between  $k \in \mathbb{N}$  thresholds, the attacker's expected utility for playing the  $(n-1)^{\text{th}}$  threshold still must be equal to  $r_a(t_n)$ . It implies the fact that the  $(n-1)^{\text{th}}$  threshold must remain uncovered with the computed probability. The defender can cover the  $(n-1)^{\text{th}}$  threshold only by playing the  $n^{\text{th}}$  threshold. And so, the computed defender's probability  $p_n$  of playing the  $n^{\text{th}}$  threshold remains the same in the case of  $k$  thresholds.

When in the second step of the algorithm the defender makes three lowest thresholds equally appealing to the attacker, he can compute how often the  $(n - 2)^{th}$  threshold must be unprotected using an analogous equation

$$r_a(t_n) = p_{n-2,3} \cdot r_a(t_{n-2}) + (1 - p_{n-2,3}) \cdot p_a.$$

As a result,

$$p_{n-2,3} = \frac{r_a(t_n) + p_a}{r_a(t_{n-2}) + p_a}. \quad (5.2.7)$$

Probability that the  $(n - 1)^{th}$  threshold is uncovered is equal to  $(p_{n-1,3} + p_{n-2,3})$ . The  $(n - 1)^{th}$  threshold must be equally appealing to the attacker as in the case of two thresholds. Hence, the defender's probability of playing the  $(n - 1)^{th}$  threshold  $p_{n-1} = p_{n-1,2} - p_{n-2,3}$ . On the set of reasonable thresholds the attacker's reward function  $r_a$  is non-decreasing. Taking this fact into account together with formulas for  $p_{n-1,2}$  and  $p_{n-2,3}$ , see Eqs.5.2.6, 5.2.7, it is obvious that  $p_{n-1,2} \geq p_{n-2,3}$ . Therefore,  $p_{n-1}$  can be always computed using the described procedure.

Probability  $p_{n-2,3}$  of leaving the  $(n-2)^{th}$  uncovered has to remain the same even when more thresholds are made equally appealing to the attacker. Consequently, probability  $(p_n + p_{n-1})$  of covering the  $(n-2)^{th}$  threshold must remain the same. As  $p_n$  is constant no matter how many thresholds are taken into consideration by the defender, the computed probability  $p_{n-1}$  is constant in the same way. Further probabilities  $p_{n-2}, p_{n-3}, \dots, p_1$  can be computed analogously to the computation of  $p_{n-1}$ . The overall algorithm is summed up as Algorithm 2.

**Input:** ordinal number  $N$  of the lowest threshold to consider,  $r_a, p_a$

**Output:** the defender's NE strategy  $\varsigma$

```

1  $i = N - 1;$ 
2  $\varsigma_N = 1;$ 
3 while  $i > 0$  do
4    $\text{weighting\_factor} = \frac{r_a(t_N) + p_a}{r_a(t_i) + p_a};$ 
5    $\varsigma_{i+1} = \varsigma_{i+1} - \text{weighting\_factor};$ 
6    $\varsigma_i = \text{weighting\_factor};$ 
7    $i = i - 1;$ 
8 end
9 return  $\varsigma;$ 

```

**Algorithm 2:** The algorithm for determining the defender's NE strategy.

### 5.3. Stackelberg equilibrium

If an attacker can estimate the defender's strategy before acting, then NE should not be used as a solution concept. SE concept suits better the case when a defender must come up with an optimal randomized strategy knowing that later on an attacker will response optimally to the strategy [66]. We devise the quadratic time algorithm for computation of the SSE in the model.

### 5.3.1. Computation of the players' strategies

When dealing with a normal-form game, it is possible to find Stackelberg strategy by formulating for every threshold  $t_a$  a related LP-problem [75]:

$$\max_{\mathbf{d}} - \sum_{t=t_1}^{t_n} d_t c_d^b(t) - c_{t_a}^r \sum_{t=t_a}^{t_n} d_t; \quad (5.3.1a)$$

$$d_t \in [0, 1] \quad \forall t \in \{t_1, \dots, t_n\} \quad (5.3.1b)$$

$$\sum_{t=t_1}^{t_n} d_t = 1 \quad (5.3.1c)$$

$$U_a(t_a, \mathbf{d}) \geq U_a(t, \mathbf{d}) \quad \forall t \in \{t_1, \dots, t_n\}. \quad (5.3.1d)$$

The LP says that the defender looks for a mixed strategy  $\mathbf{d}$  which would maximize his utility, or in other words, would minimize his overall costs.  $d_t$  denotes probability of playing threshold  $t$  as a part the mixed strategy  $\mathbf{d}$ . Let  $t_n$  and  $t_1$  denote the lowest and the highest thresholds from  $\mathbf{T}$  correspondingly,  $t_n = \min\{\mathbf{T}\}$ ,  $t_1 = \max\{\mathbf{T}\}$ . In objective 5.3.1a the first summand  $\sum_{t=t_1}^{t_n} d_t c_d^b(t)$  corresponds to expected costs due to background costs and the second summand  $c_{t_a}^r \sum_{t=t_a}^{t_n} d_t$  represents expected costs due to attack by the rational attacker on the threshold  $t_a$ . Constraint 5.3.1d ensures that the attacker plays a best response to the defender's strategy  $\mathbf{d}$ . From the form of constraint 5.3.1d you may note that we use the SSE assumption, see Section 3.1.4 for more details.

By solving the LP's, a set of strategy profiles would be obtained. One can see from LP 5.3.1 that the resulting strategy profile  $(\mathbf{d}, t_a)$  satisfies all conditions for SSE except for the condition that the defender forces an attacker to play a threshold which maximizes the defender's overall utility. Hence, in order to get the defender's optimal SSE strategy we must choose a solution to the LP, which corresponds to the highest value of the defender's utility.

In this work each LP subproblem is not solved using a standard solver. Rather we construct a solution to each LP subproblem in linear time. The main idea is to gradually expand a set of targets equally appealing to the attacker. The set of all targets equally appealing to the attacker is called an attack set.

Note that the approach of gradual attack set expansion was used in ORIGAMI algorithm to compute SSE in a normal-form game optimizing allocation of security resources in such real-world applications as scheduling the police patrolling or randomized baggage screening [43]. The ORIGAMI algorithm cannot be used to solve the problem formulated in this work, as there are several principal differences between the problem solved with ORIGAMI and the problem we face. In the addressed by ORIGAMI problem it is possible to put additional protection on one particular target without affecting any other target. On the other hand, in the case of the formulated adversarial classification game it is impossible to strengthen the protection of a threshold independently on protection of higher thresholds. To be specific, if we start detecting instances with danger probability exceeding a threshold  $t$ , then we automatically start detecting instances with danger probability exceeding any  $t' > t$  as well. Considering the ORIGAMI, it is assumed that the attack set expansion cannot provide a worse outcome for the defender. On contrary, in our case there are additional background costs for protection of each target, i.e., each threshold. Hence, before any expansion of the attack set it is needed to determine if protection of a corresponding threshold improves the defender's utility or not.

As we have already formalized it in LP 5.3.1, the attacker chooses a threshold which



guarantees to him the highest possible utility with respect to a given defender's strategy. In each LP subproblem we fix the threshold an attacker would choose. It can be shown that there exists just one pure strategy of the defender such that the fixed attacker's threshold belongs to the attack set.

**Proposition 5.3.1.** *For any fixed threshold  $t_a$  there is only one defender's pure strategy which satisfies constraint 5.3.1d. The pure strategy for the defender is to play  $t_a$ .*

**Proof:** The proposition can be proved by contradiction. Let us assume there is another defender's pure strategy which satisfies constraint 5.3.1d. It can not be any threshold lower than the fixed threshold  $t_a$ , as if the defender plays a lower threshold then the attacker's utility for playing  $t_a$  is a cost for being detected. However, for playing the lower threshold, which the defender plays, the attacker would achieve a non-negative payoff, as the attacker's reward function  $r_a(t)$  is non-negative and the attacker would not be detected.

At the same time, a threshold  $t_h$  higher than  $t_a$  can correspond to the attacker's reward  $r_a(t_h) > r_a(t_a)$ . Therefore, if the defender plays  $t_h$ , making it unprotected, constraint 5.3.1d does not hold in general, because the attacker might prefer to play  $t_h$ .

This contradicts the assumption that there is a defender's pure strategy another than  $t_a$  which satisfies constraint 5.3.1d. ■

It can be shown that the defender's pure strategy from Prop. 5.3.1, i.e., playing the fixed attacker's threshold  $t_a$ , is not a solution to LP 5.3.1 in general.

**Proposition 5.3.2.** *In case when there is at least one threshold  $t_h$  higher than  $t_a$  and for background costs holds  $c_d^b(t_a) > c_d^b(t_h)$ , then there is a defender's mixed strategy satisfying constraint 5.3.1d with a higher expected utility compared to the defender's utility of always playing  $t_a$ .*

**Proof:** To put the proposition differently, it might be possible to maximize the defender's expected utility by expanding the attack set  $\{t_a\}$  to higher thresholds. Expansion of the attack set means including new thresholds into the set of thresholds appealing to the attacker. The expansion of the attack set  $\{t_a\}$  to higher thresholds effectively means starting playing some higher thresholds with non-zero probability. In terms of the utility  $U_d = -\sum_{t=t_1}^{t_n} d_t c_d^b(t) - c_a^r(t_a) \sum_{t=t_a}^{t_n} d_t$  the expansion of the initial attack set to higher thresholds leaves a value of the summand  $c_a^r(t_a) \sum_{t=t_a}^{t_n} d_t$  unchanged. Note that  $\sum_{t=t_a}^{t_n} d_t$  is the probability that the threshold  $t_a$  is not covered by the defender. In other words, the expansion to higher thresholds aims to minimize background costs  $\sum_{t=t_1}^{t_n} d_t c_d^b(t)$  while not changing probability of leaving  $t_a$  uncovered.

Let us discuss in more details an attack set expansion to thresholds higher than  $t_a$ . It is required to analyze under which conditions such expansion would not violate constraint 5.3.1d.

In case of the defender's pure strategy of playing  $t_a$ , any higher threshold  $t_h$  was always covered by the defender, i.e., the attack on  $t_h$  was always detected. However, if instead of always playing  $t_a$  the defender starts playing  $t_h$  and higher thresholds with non-zero probability  $p_{unprotected}(t_h) > 0$ , then with the probability  $p_{unprotected}(t_h)$  the attack on  $t_h$  would be undetected. In order to ensure that constraint 5.3.1d is satisfied, the defender must not leave any  $t_h$  with too weak protection. Informally, the protection is too weak when the defender plays thresholds not lower than  $t_h$  with too high probability. What can happen in this case is that a weakly protected higher threshold would

## 5. Efficient Algorithms for Optimal Solutions

be more appealing to the attacker than the fixed threshold  $t_a$ . This would contradict constraint 5.3.1d. As a result, there exists a maximum allowed probability of leaving any higher threshold uncovered. The maximum probability can be computed based on the fixed threshold attacker's reward  $r_a(t_a)$  and the fact that expected utility for playing any threshold must be not higher than  $r_a(t_a)$ . Based on Eq. 5.2.7, the maximum probability of leaving a particular threshold unprotected can be computed as

$$p_{\text{unprotected,max}}(t_h) = \frac{r_a(t_a) + p_a}{r_a(t_h) + p_a} \quad (5.3.2)$$

To sum up, in order to minimize the total background costs for leaving the fixed attacker's threshold  $t_a$  uncovered, the defender has incentive to play higher thresholds with background costs  $c_d^b(t) < c_{t_a}^b$ . However, each higher threshold  $t_h$  can be uncovered at most with the probability  $p_{\text{unprotected,max}}(t_h)$ , in order to satisfy constraint 5.3.1d.

Furthermore, the attacker's reward function for a successful attack is non-decreasing on thresholds. From this fact and from Eq. 5.3.2 it follows that the higher the threshold is, the lower the maximum probability of allowed non-protection  $p_{\text{unprotected,max}}(t_h)$  is. Formally,  $\forall t_i, t_j : t_i < t_j \implies p_{\text{unprotected,max}}(t_i) \geq p_{\text{unprotected,max}}(t_j)$ . Therefore, for any threshold  $t_h$  higher than the fixed one  $t_a$ ,  $\forall t_h : t_a < t_h \implies p_{\text{unprotected,max}}(t_a) \geq p_{\text{unprotected,max}}(t_h)$ . As a result, if a threshold  $t_h$  is higher than  $t_a$ , then it is always possible to start playing the threshold  $t_h$  with probability  $p(t_h) = \frac{r_a(t_a) + p_a}{r_a(t_h) + p_a}$  and play the fixed threshold  $t_a$  with probability  $p(t_a) = 1 - p(t_h) \geq 0$ . This way constraint 5.3.1d would not be violated and at the same time the defender's expected utility is higher compared to the utility of the pure strategy from Prop. 5.3.1, as  $c_d^b(t_a) > c_d^b(t_h)$  according to the assumptions. ■

In the proof of Proposition 5.3.2 we analyzed some properties of the defender's mixed strategies on thresholds non-lower than the fixed threshold  $t_a$ . Furthermore, based on the analyzed properties and constraint 5.3.1d it is possible to show that some of the thresholds higher than  $t_a$  are dominated. This leads to a result similar to Prop. 5.2.1 which holds for NE on all thresholds. In the following proposition for the most of dominated actions a concept of strict domination applies, see Section 3.1.3 for definitions regarding the domination. Moreover, we use the concept of the weak domination as well. If an action of choosing one threshold is no worse than choosing another threshold, we prefer choosing the threshold providing stronger protection. Remember, the lower the threshold is, the more malicious instances are detected, and therefore the stronger the protection is.

**Proposition 5.3.3.** *When satisfying constraint 5.3.1d, the defender cannot improve his utility by playing a threshold  $t_d$  instead of playing  $t'_d$  if  $t_d > t'_d \wedge c_d^b(t_d) \geq c_d^b(t'_d)$ , where  $t_d$  and  $t'_d$  are thresholds higher than the fixed attacker's threshold  $t_a$ .*

**Proof:** It is given by the fact that under the satisfied constraint 5.3.1d the attacker always plays the fixed threshold  $t_a$ . Therefore, the costs due to the rational attacker is always the same. Therefore, background costs are the only costs which differ for playing different thresholds. Thus, for the defender it is better to play thresholds with lower background costs. Note that in the discussed case when  $t_d > t'_d \wedge c_d^b(t_d) \geq c_d^b(t'_d)$  the defender can start playing the smaller-cost threshold  $t'_d$  instead of playing the higher-cost threshold  $t_d$  without violation of constraint 5.3.1d, see proof to the Prop. 5.3.2 for more details. ■

Note that in case  $t_d > t'_d \wedge c_d^b(t_d) < c_d^b(t'_d)$  it is in general not possible to move the whole probability of playing a higher-cost threshold  $t'_d$  onto the lower-cost threshold  $t_d$  without violation of constraint 5.3.1d. This is given by the fact that the maximum allowed probability of leaving the higher threshold  $t'_d$  unprotected is lower or equal to the analogous maximum probability for the threshold  $t_d$ .

It has been shown that constraint 5.3.1d leads to domination of some of the defender's thresholds. Capitalizing on this result, it is possible to compute an optimal defender's mixed strategy on non-dominated thresholds higher or equal to the fixed threshold.

**Proposition 5.3.4.** *On non-dominated thresholds higher and equal to the fixed threshold  $t_a$  the defender's mixed strategy obtained with Alg. 2 is a solution to LP 5.3.1 on this set of thresholds, i.e., it is optimal in terms of the defender's expected utility and does not violate constraint 5.3.1d regarding the attack set<sup>2</sup>.*

**Proof:** The Alg. 2 leaves each threshold unprotected with a maximum allowed probability from Eq. 5.3.2. Therefore, the attack set condition from LP 5.3.1 is satisfied by construction.

At the same time, the defender's background costs are minimized, as by construction of the mixed strategy the threshold corresponding to the lowest background costs is played with the maximum probability which does not violate the attack set condition. Note that the threshold corresponding to the lowest background costs is the highest one from the defender's support. The threshold just before the lowest-cost threshold is played with a remaining probability, i.e., the probability which cannot be moved to the lowest-cost threshold in order to not violate the attack set condition from LP 5.3.1. At the same time it is the highest possible probability to play the second lowest-cost threshold, derived from the maximum allowed probability to leave the threshold unprotected. As a result, for the second and the third highest thresholds in the support it holds that no probability can be moved on threshold(-s) with lower background costs without violating constraint 5.3.1d. As the mixed strategy is constructed analogously for all the thresholds, for each threshold it holds that without violation of the attack set constraint 5.3.1d, it is impossible to reduce background costs. In other words, on thresholds higher or equal to the fixed threshold  $t_a$  the computed with Alg. 2 mixed strategy corresponds to minimal expected background costs.

Moreover, the defender's costs for attacks by the rational attacker is the same for all mixed strategies satisfying constraint 5.3.1d, as the attacker always plays the fixed threshold  $t_a$ . As a result, the mixed strategy satisfying constraint 5.3.1d and corresponding to minimal background costs is optimal in terms of the defender's expected utility. ■

The mixed strategy from Prop. 5.3.4 is optimal on thresholds higher or equal to the fixed thresholds  $t_a$ . However, the strategy might not be optimal in case there is a lower threshold with low background costs.

**Proposition 5.3.5.** *If there is a threshold  $t_l$  lower than  $t_a$ ,  $t_l < t_a$ , with background costs  $c_d^b(t_l)$  lower than the defender's expected costs for the mixed strategy from Prop. 5.3.4, then the mixed strategy is never optimal solution to LP 5.3.1.*

<sup>2</sup>Remember, Alg. 2 requires the lowest threshold to consider as an input. The fixed threshold  $t_a$  is thus the required input.

**Proof:** Informally, in this case a defender can improve his expected utility by starting playing the lower threshold  $t_l$  with such probability, that the attack set constraint 5.3.1d still would be satisfied.

The proof is structured as follows: first, it is proven that the defender can always start playing  $t_l$  without violation of constraint 5.3.1d. The allowed probability  $d(t_l)$  of playing  $t_l$  is derived. Next, it is shown how to derive the defender's mixed strategy which would satisfy constraints of LP 5.3.1 while a support of the strategy would include  $t_l$  and the support of the mixed strategy from Prop. 5.3.4. Finally, it is shown that the expected utility of the newly derived mixed strategy is higher than the expected utility of the strategy from Prop. 5.3.4.

a) *Playing  $t_l$  without violation of constraint 5.3.1d.*

By starting playing  $t_l$ , the defender puts some protection on higher thresholds including  $t_a$ . The defender must not put too much protection on the fixed threshold  $t_a$ , as the fixed threshold must still belong to the attack set. The attacker's expected utility for playing the fixed threshold must be higher or equal to the attacker's reward for playing the threshold  $t_l$ . Note that the attack on the lower threshold  $t_l$  is always undetected. Thus,

$$r_a(t_l) \leq (1 - d(t_l)) \cdot r_a(t_a) - p_a \cdot d(t_l) \implies d(t_l) \leq \frac{r_a(t_a) - r_a(t_l)}{r_a(t_a) + p_a}.$$

To sum up, when satisfying the attack set constraint 5.3.1d, the defender can start playing the lower thresholds  $t_l$  at most with probability

$$d_{max}(t_l) = \frac{r_a(t_a) - r_a(t_l)}{r_a(t_a) + p_a}. \quad (5.3.3)$$

b) *The defender's mixed strategy satisfying the constraints of LP 5.3.1*

As a result of playing  $t_l$  with probability  $d_{max}(t_l)$ , a total probability left for playing thresholds equal to or higher than the fixed threshold is  $(1 - d_{max}(t_l))$ . In other words,  $(1 - d_{max}(t_l))$  is probability of leaving the fixed threshold unprotected. Again, we aim to minimize the defender's background costs for the given total probability of leaving  $t_a$  unprotected.

Analogously to Eq. 5.3.2, it is possible to determine a maximum probability of playing a threshold  $t_h$ ,  $t_h > t_l$ , without violation of constraints from LP 5.3.1:

$$p'_{unprotected,max}(t_h) = \frac{r_a(t_l) + p_a}{r_a(t_h) + p_a}.$$

Moreover,  $r_a(t_l) = (1 - d_{max}(t_l)) \cdot r_a(t_a) - p_a \cdot d_{max}(t_l)$ . Therefore,  $r_a(t_l) + p_a = (1 - d_{max}(t_l)) \cdot (r_a(t_a) + p_a)$ .

As a result, a newly computed maximum allowed probability of leaving the threshold  $t_h$  unprotected,  $p'_{unprotected,max}(t_h)$ , can be easily obtained using the probability  $p_{unprotected,max}(t_h)$  of playing the corresponding threshold in the optimal mixed strategy from Prop. 5.3.4, computed using Eq. 5.3.2:

$$\begin{aligned} p_{unprotected,max}(t_h) &= \frac{r_a(t_a) + p_a}{r_a(t_h) + p_a} \wedge r_a(t_l) + p_a = (1 - d_{max}(t_l)) \cdot (r_a(t_a) + p_a) \wedge \\ &\wedge p'_{unprotected,max}(t_h) = \frac{r_a(t_l) + p_a}{r_a(t_h) + p_a} \implies p'_{unprotected,max}(t_h) = \\ &= (1 - d_{max}(t_l)) p_{unprotected,max}(t_h). \end{aligned}$$

Therefore, we can elegantly compute an optimal<sup>3</sup> mixed strategy with the support including the lower threshold  $t_l$ , the fixed threshold  $t_a$  and all non-dominated thresholds higher than  $t_a$ . Probability of playing  $t_l$  is  $d_{max}(t_l)$ , and in order to determine probability of playing any other threshold  $t'$ , it is sufficient to multiply the probability of playing  $t'$  in the optimal mixed strategy from Prop. 5.3.4 by  $(1 - d_{max}(t_l))$ .

*c) The expected utility of the derived mixed strategy*

We derived the mixed strategy with the support including  $t_l$  and thresholds from the support of the mixed strategy from Prop. 5.3.4. Knowing a player's mixed strategy and his utility function it is straightforward to compute the expected utility. To make the discussion more intuitive, we consider expected costs of the derived mixed strategy, which is the negated expected utility.

There is an intuitive view on the expected costs of the derived mixed strategy. The lower threshold is played with probability  $d_{max}(t_l)$  and all thresholds belonging to the support of the optimal mixed strategy from Prop. 5.3.4 are played with probabilities  $p(t) = (1 - d_{max}(t_l)) \cdot p_{unprotected,max}(t)$ , where  $p_{unprotected,max}(t)$  is probability of playing  $t$  in the optimal mixed strategy on thresholds not lower than  $t_a$ . Thus, if we view choosing to play the mixed strategy from Prop. 5.3.4 as one action available to the defender, and playing the lower threshold  $t_l$  as another available action, then the defender chooses to play  $t_l$  with probability  $d_{max}(t_l)$  and stick to the previously computed mixed strategy with probability  $(1 - d_{max}(t_l))$ . The expected costs is then

$$C_{new} = d_{max}(t_{new})c_d^b(t_l) + (1 - d_{max}(t_{new}))C_0,$$

where  $C_0$  stands for the expected costs for the optimal mixed strategy from Prop. 5.3.4. The attacker's reward function is non-decreasing, hence,

$$t_l < t_a \implies r_a(t_a) \geq r_a(t_l) \xrightarrow{Eq.5.3.3} d_{max}(t_l) \geq 0 \xrightarrow{c_d^b(t_l) < C_0} C_{new} \leq C_0. \blacksquare$$

In the proof of the Prop. 5.3.5, in order to show that the optimal mixed strategy on thresholds not lower than  $t_a$  might not be a solution to LP 5.3.1 in general, we constructed the mixed strategy corresponding to better defender's expected utility. The constructed mixed strategy is moreover optimal on the considered thresholds.

**Proposition 5.3.6.** *Let  $t_l$  be a threshold lower than  $t_a$  with background costs  $c_d^b(t_l)$  lower than the total defender's expected costs for the mixed strategy from Prop. 5.3.4. The optimal defender's mixed strategy on the set of thresholds  $\{t_l, t_a, \text{all non-dominated thresholds not-lower than } t_a\}$  is the one constructed in the proof of Prop. 5.3.5.*

**Proof:** Note that the mixed strategy on the set of thresholds was in fact obtained with Alg. 2 and satisfies constraint 5.3.1d regarding the attack set. Therefore, the proof is analogous to the proof of Prop. 5.3.4.  $\blacksquare$

In Props. 5.3.5,5.3.6 it was not discussed whether there are any thresholds between  $t_a$  and  $t_l$ . In case there are such thresholds, it can be shown that some of them are dominated.

<sup>3</sup>For more details on the optimality of the mixed strategy see Prop. 5.3.6.

**Proposition 5.3.7.** *When satisfying constraint 5.3.1d, the defender cannot improve his utility by playing a threshold  $t_d$  instead of playing  $t'_d$  if  $t_d > t'_d \wedge c_d^b(t_d) \geq c_d^b(t'_d)$ , where  $t_d$  and  $t'_d$  are thresholds lower than the fixed attacker's threshold  $t_a$ .*

**Proof:** When the attack set condition of LP 5.3.1 is satisfied, the attacker always plays the fixed threshold. Moreover, considering playing one threshold lower than  $t_a$  instead of another threshold lower than  $t_a$ , probability of covering the  $t_a$  always remains the same. As a result, the defender's costs due to attacks by the rational attacker is always the same. Having said that, the proof is analogous to the proof of Prop. 5.3.3. ■

It has been shown that thresholds lower than  $t_a$  can improve the defender's expected utility. Furthermore, it has been shown that some thresholds might be dominated. However, it has not been discussed under which conditions a non-dominated threshold lower than  $t_a$  is guaranteed to belong to the support of an optimal defender's mixed strategy.

**Proposition 5.3.8.** *Let  $\mathbf{d}$  be the optimal defender's strategy on thresholds equal to or higher than a threshold  $t_0$ ,  $t_0 \leq t_a$ . Let  $t_l$  be a non-dominated threshold lower than  $t_0$ . If background costs  $c_d^b(t_l)$  are lower than the defender's expected costs  $C(\mathbf{d})$  for playing the mixed strategy  $\mathbf{d}$  and  $t_l$  is the highest threshold which satisfies this property, denoted  $t_l = \max\{t \mid c_d^b(t) < C(\mathbf{d}) \wedge t \text{ is non-dominating}\}$ , then on thresholds higher or equal to  $t_l$ ,  $t_l$  belongs to the support of an optimal defender's mixed strategy solving LP 5.3.1.*

**Proof:**

From Prop. 5.3.6 it follows that  $t_l$  belongs to an optimal mixed strategy on the following set of thresholds:  $\{t_l, t_h \mid t_h \geq t_0 \wedge t_h \text{ is non-dominating}\}$ . Moreover, there are no non-dominated thresholds between  $t_l$  and  $t_0$  satisfying the property on background costs, as  $t_l = \max\{t \mid c_d^b(t) < C(\mathbf{d}) \wedge t \text{ is non-dominating}\}$ . Therefore,  $t_l$  belongs to the optimal defender's mixed strategy on thresholds  $\{t_h \mid t_h \geq t_l \wedge t_h \text{ is non-dominating}\}$ . ■

It is worth discussing the condition on optimality of the mixed strategy  $\mathbf{d}$  from Prop. 5.3.8. Let us show why the strategy  $\mathbf{d}$  must be an optimal one on all thresholds higher than or equal to  $t_0$ , otherwise  $t_l$  might not belong to the optimal mixed strategy on thresholds higher than or equal to  $t_l$ . Let  $C(\mathbf{d})$  denote the defender's expected costs associated with the mixed strategy  $\mathbf{d}$ . Assume, the strategy  $\mathbf{d}$  is not optimal on thresholds higher than or equal to  $t_0$ , and it holds that  $C(\mathbf{d}) > c_d^b(t_l)$ . If  $\mathbf{d}$  is not an optimal strategy, then there exists a defender's strategy  $\mathbf{d}'$  such that the corresponding defender's expected costs  $C(\mathbf{d}) > C(\mathbf{d}')$ . Nothing forbids a situation when  $C(\mathbf{d}) > c_d^b(t_l) > C(\mathbf{d}')$ , when  $t_l$  does not belong to the support of the optimal defender's mixed strategy on thresholds higher than or equal to  $t_l$ .

It means that it is impossible to expand the attack set to thresholds lower than  $t_a$  until the defender's expected costs on all thresholds greater or equal to  $t_a$  is minimized.

Next, even when the strategy  $\mathbf{d}$  is optimal on thresholds  $\{t \mid t \geq t_0\}$  and  $C(\mathbf{d}) > c_d^b(t_l)$ , it might happen that there is a threshold  $t'_l > t_l$  such that  $C(\mathbf{d}) > c_d^b(t'_l)$ . That might lead to a mixed strategy on thresholds  $\{t \mid t \geq t'_l\}$  which is more appealing to the defender than playing  $t_l$ . Therefore, the expansion of the attack set to lower thresholds must be done without skipping the maximal threshold  $t_l = \max\{t \mid c_d^b(t) < C(\mathbf{d}) \wedge t \text{ is non-dominating}\}$ .

Note that the proven facts enable us to derive linear time algorithm for computing the solution to LP 5.3.1. First, based on Props. 5.3.3, 5.3.7 we filter out all dominated thresholds. Then, using Alg. 2 we compute an optimal mixed strategy on thresholds higher or equal to the fixed threshold  $t_a$ . Next, while there are thresholds satisfying the conditions from Prop. 5.3.8, i.e.,  $Z = \{t \mid c_d^b(t) < C(\mathbf{d}) \wedge t \text{ is non-dominated}\} \neq \emptyset$ , we add the highest threshold from the set  $Z$  to the attack set. It is done analogously to the procedure from the proof to Prop. 5.3.5. Namely, the probability of playing the newly added threshold is computed as:

$$d(t_{new}) = \frac{r_a(t_{prev}) - r_a(t_{new})}{r_a(t_{prev}) + p_a},$$

where  $t_{new}$  is the next threshold to add to the attack set,  $t_{prev}$  is the lowest threshold in the attack set before adding of  $t_{new}$ .

Therefore, if the total expected utility was  $U_{previous}$  before, then it becomes

$$U_{new} = -d(t_{new})c_d^b(t_{new}) + (1 - d(t_{new}))U_{previous}.$$

This enables us to determine the lowest threshold of the defender's support. After that, knowing the lowest threshold of the optimal attack set, the construction of the defender's optimal strategy again can be done using Alg. 2. The algorithm for computation of the solution to LP 5.3.1 is summarized in Alg. 3<sup>4</sup>. Note that Alg. 2 constructs an optimal strategy on the support consisted of all non-dominated thresholds. If you compare Prop. 5.2.1 regarding dominated actions in case of NE and Props. 5.3.3, 5.3.7, the only difference between the sets of non-dominated thresholds might arise due to  $t_a$ , which must belong to the support in case of LP 5.3.1, because otherwise constraint 5.3.1d would be violated.

**Input:**  $t_a, r_a, p_a$

**Output:** the defender's strategy  $\mathbf{d}$

- 1 Filter out dominated thresholds based on Props. 5.3.3, 5.3.7;
- 2  $\mathbf{d}_h = \text{Alg. 2}(t_a, r_a, p_a)$ ;
- 3  $U =$  expected costs of  $\mathbf{d}_h$ ;
- 4  $t_{prev} = t_a$ ;
- 5 **while**  $\{t \mid t < t_{prev} \wedge c_d^b(t) < U\} \neq \emptyset$  **do**
- 6      $t = \max\{t \mid t < t_{prev} \wedge c_d^b(t) < U\}$ ;
- 7      $d(t) = \frac{r_a(t_{prev}) - r_a(t)}{r_a(t_{prev}) + p_a}$ ;
- 8      $U = d(t)c_d^b(t) + (1 - d(t))U$ ;
- 9      $t_{prev} = t$ ;
- 10 **end**
- 11  $\mathbf{d} = \text{Alg. 2}(t_{prev}, r_a, p_a)$ ;
- 12 **return**  $\mathbf{d}$ ;

**Algorithm 3:** The algorithm for computation the defender's strategy for LP 5.3.1.

Considering the complexity of the algorithm, it is obvious that it takes three passes through all thresholds in the worst case, when the fixed attacker's threshold is the highest threshold and the lowest threshold belongs to the SSE support as well. During each

<sup>4</sup>To be precise, as a first input to Alg. 2 it is required to provide an order number of the lower threshold to consider. For the sake of conciseness in the pseudocode we directly provide a threshold as an input, but the threshold's order number is meant.

pass we perform just simple constant-time operations on thresholds. Hence, construction of the solution to one subgame is done in linear time. It in its turn leads to the quadratic-time algorithm for computation of the defender’s SSE strategy.

## 5.4. FPR Restriction

Restriction on FPR is an important requirement in practical applications of adversarial classification [5, 35, 24, 13, 39, 29, 47, 28]. For that reason we suggest how to devise a randomized adversary-aware classification respecting the restriction on FPR.

The most naive approach to restrict FPR is to set the lowest classification threshold with corresponding FPR not exceeding the maximum expected FPR, if the optimal classification threshold violates the FPR constraint. However, in this approach adaptivity of the adversaries is ignored leading to the problems discussed in Sections 1.1, 2.3. A better straightforward solution is to choose a fixed classification threshold optimizing the adversary-aware utility  $U_d^u(t_d)$  under the restriction on maximum expected FPR. However, this solution does not leverage from randomization which was found to mitigate the capabilities of adversaries [36]. Another option is to consider all thresholds corresponding to FPR’s not exceeding the maximum expected FPR and devise a randomized equilibrium strategy on those thresholds only. Due to the fact that the fixed threshold optimizing  $U_d^u(t_d)$  also belongs to the considered thresholds, the defender’s utility cannot be lower than the utility for the fixed threshold, analogously to the result from [36].

Even though the latter approach makes it possible to devise a randomized adversary-aware security, disregarding thresholds limits abilities of the defender significantly. We rather analyze computation of the randomized security without limiting ourselves to particular thresholds.

We suggest an approximation algorithm for computation of SSE under the restriction on the expected FPR. First, we analyze a modified version of the optimization problem 5.3.1 for SSE computation with constraint on FPR introduced. We show how to check whether or not there is a feasible solution to the FPR-constrained LP when a threshold  $t$  is the lowest one from the attack set<sup>5</sup>. Based on the discovered result we develop an approximation algorithm for computation of SSE. Lastly, we show how to derive an upper bound on the utility of the optimal solution, which enables estimation of the absolute performance guarantee for the solution provided with the algorithm.

### 5.4.1. SSE computation

First, we analyze the computation of the SSE under the restriction on maximum expected FPR. The high-level idea of the SSE computation remains the same: for each possible attacker’s fixed threshold  $t_a$  resolve LP analogous to LP 5.3.1. Let  $\phi_t$  denote a FPR<sup>6</sup> corresponding to the threshold  $t$ . Let  $\Phi^{max}$  denote the maximum expected FPR which can be tolerated in a specific security application. Let  $\mathbf{d}$  denote the defender strategy. Then the introduced restriction on expected FPR is formulated as  $\sum_{t=t_1}^{t_n} d_t \phi_t \leq \Phi^{max}$ . A modified LP is summarized in LP 5.4.1.

<sup>5</sup>Remember that the Alg. 3, which computes an optimal solution to LP 5.3.1 without the FPR-constraint, gradually added thresholds lower than the fixed threshold  $t_a$  to the attack set if it improved the defender’s utility. Lower thresholds are assumed to correspond to weaker attacks, see Section 4.1.4.

<sup>6</sup>FPR’s are assumed to be estimated based on the classifier ROC curve.



$$\max_{\mathbf{d}} - \sum_{t=t_1}^{t_n} d_t c_d^b(t) - c_{t_a}^r \sum_{t=t_a}^{t_n} d_t; \quad (5.4.1a)$$

$$d_t \in [0, 1] \quad \forall t \in \{t_1, \dots, t_n\} \quad (5.4.1b)$$

$$\sum_{t=t_1}^{t_n} d_t = 1 \quad (5.4.1c)$$

$$U_a(t_a, \mathbf{d}) \geq U_a(t, \mathbf{d}) \quad \forall t \in \{t_1, \dots, t_n\} \quad (5.4.1d)$$

$$\sum_{t=t_1}^{t_n} d_t \phi_t \leq \Phi^{max}. \quad (5.4.1e)$$

Capitalizing on previous findings, we develop a procedure to solve LP 5.4.1. From Section 5.3 it is known that the mixed strategy  $\mathbf{d}$  obtained with  $\mathbf{d} = \text{Alg. } 3(t_a, r_a, p_a)$  is the optimal defender's solution to LP 5.3.1. LP 5.3.1 is exactly LP 5.4.1 with constraint 5.4.1e relaxed. If for  $\mathbf{d}$  5.4.1e is satisfied, then the computed strategy is also an optimal solution to LP 5.4.1.

Let us analyze the case when the mixed strategy  $\mathbf{d} = \text{Alg. } 3(t_a, r_a, p_a)$  violates constraint 5.4.1e. Remember that Alg. 3 gradually extended the attack set by including lower thresholds to the set if it improved the defender's utility. Extension of the attack set to lower threshold can be interpreted as forcing the adversary to consider a weaker attack<sup>7</sup>. Let the lowest threshold belonging to the  $\mathbf{d}$ -strategy attack set be denoted  $t_{weakest\ attack}$ . Note that from the discussion in Section 5.3 and Alg. 3 it follows that  $t_{wa}$  is the lowest threshold from the support of  $\mathbf{d}$ .

We introduce a method how to check whether there is a feasible defender's strategy, satisfying both 5.4.1d and 5.4.1e and forcing the adversary to consider the attack on the threshold  $t_{wa}$ .

**Proposition 5.4.1.** *Consider a problem of constructing the defender's strategy satisfying 5.4.1e for the threshold  $t_{wa}$  and minimizing the expected FPR. The problem can be formalized as LP 5.4.2.*

$$\min_{\mathbf{d}} \sum_{t=t_1}^{t_n} d_t \phi_t; \quad (5.4.2a)$$

$$d_t \in [0, 1] \quad \forall t \in \{t_1, \dots, t_n\} \quad (5.4.2b)$$

$$\sum_{t=t_1}^{t_n} d_t = 1 \quad (5.4.2c)$$

$$U_a(t_{wa}, \mathbf{d}) \geq U_a(t, \mathbf{d}) \quad \forall t \in \{t_1, \dots, t_n\} \quad (5.4.2d)$$

$\mathbf{d}_\phi = \text{Alg. } 2(t_{wa}, r_a, p_a)$  is an optimal solution to LP 5.4.2 if in Alg. 2 all thresholds  $\{t \mid t \geq t_{wa}\}$  are considered, including dominated ones.

**Proof:** Alg. 2( $t_{wa}, r_a, p_a$ ) was designed to make all thresholds  $\{t \mid t \geq t_{wa}\}$  equally appealing to the attacker. Thus 5.4.2d is satisfied for  $\mathbf{d}$  by construction of the strategy.

<sup>7</sup>As it was discussed in Section 4.1.4, the assumption from [36] that the higher the classification threshold the attacker successfully bypasses, the higher the reward for undetected attack he gains, can be motivated with the intuition, that the higher the probability of being malicious the unstopped attacker had, the more damaging behavior it established.

## 5. Efficient Algorithms for Optimal Solutions

From the derivation of Alg. 2 it follows that in the resulting strategy each threshold  $t$  is uncovered with a maximum probability  $p_{unprotected,max}$  which does not violate 5.4.2d. Remember that in Alg. 2 the probability is gradually transferred from a lower threshold to the next higher threshold. From Eq. 5.2.7 for the transferred probability it holds

$$p_{unprotected,max}(t) = \frac{r_a(t_{wa}) + p_a}{r_a(t) + p_a} \quad \xRightarrow{r_a \text{ is non-decreasing}}$$

$$\xRightarrow{r_a \text{ is non-decreasing}} \forall t_1, t_2 : t_1 < t_2 \implies p_{unprotected,max}(t_1) \geq p_{unprotected,max}(t_2).$$

To sum up, when gradually transferring the probability to higher thresholds, some probability must remain on a lower threshold in order for 5.4.2d to be satisfied. At the same time, it follows that, as the maximum possible probability is transferred onto higher thresholds, the probability remained on a lower threshold is the minimal possible. This holds for all thresholds except for the highest one, as Alg. 2 terminates once on the highest threshold a maximal possible probability was transferred.

Also note that FPR is decreasing in thresholds. Thus, the highest threshold corresponds to the lowest FPR possible. Hence, the lowest-FPR threshold is played with the maximum possible probability not violating 5.4.2d. Analogously, the next lowest-FPR threshold is also played with the maximal possible probability not violation 5.4.2d, as no probability can be transferred from any higher-FPR thresholds. The same holds for all thresholds going from the highest thresholds (i.e., from the lowest-FPR thresholds) to the lowest ones (the highest-FPR ones).■

Remember that  $\mathbf{d}$  denotes the optimal solution to LP 5.3.1 when  $t_{wa}$  is the lowest threshold from the attack set. From the proven Prop. 5.4.1 it follows that  $\mathbf{d}_\phi = \text{Alg. 2}(t_{wa}, r_a, p_a, \text{include dominated thresholds})$  corresponds to the minimum possible expected FPR on the attack set extended by the dominated thresholds. Thus, in case the expected FPR of  $\mathbf{d}_\phi$  exceeds  $\Phi^{max}$ , there is no feasible solution to LP 5.4.1 with  $t_{wa}$  belonging to the attack set.  $t_{wa}$  corresponds to the highest FPR among all thresholds from the defender's support. Let us pick as  $t_{wa} = t_{wa-1}$ , i.e., the next threshold higher than  $t_{wa}$ , and consider the defender's strategy with a new  $t_{wa}$  being the lowest threshold in the attack set. From the perspective of Alg. 3 we in fact have undone the last step of the algorithm. Based on Proposition 5.4.1 we know that the utility improvement gained by adding the threshold in the last step of Alg. 3 to an attack set leads to infeasible solution to LP 5.4.1.

We then proceed further analogously, by calling Alg. 2( $t_{wa}, r_a, p_a, \text{include dominated thresholds}$ ). In case the expected FPR again exceeds the limit, we proceed analogously until we find the attack set for which there is a feasible solution to LP 5.4.1 or until  $t_{wa} = t_a$ .

In the latter case there is no feasible solution to LP 5.4.1 with the fixed threshold  $t_a$  belonging to the attack set. As  $t_a$  must belong to the attack set by the formulation of LP 5.4.1, there is no feasible solution at all. Note that for the purpose of the SSE computation a set of LP's corresponding to all possible thresholds must be solved. Therefore, there is always a feasible solution to at least one LP with  $t_a$  being equal to the highest threshold, which corresponds to zero FPR.

Let us now consider the former case when a feasible solution to LP 5.4.1 was found. Let  $t_{wa}$  denote the lowest threshold belonging to the attack set in the feasible solution. It is known that no lower threshold can belong to the attack set, in order for LP 5.4.1 to have a solution. We are about to devise an algorithm for finding the optimal solution

to LP 5.4.1 on thresholds  $\{t \mid t \geq t_{wa}\}$ .

First, we compute an optimal solution to LP 5.3.1 on the attack set with  $t_{wa}$  being the lowest threshold from the set. It can be done straightaway. As in Alg. 3, once we know the lowest threshold  $t_{wa}$  from the attack set, we can compute the optimal solution  $\mathbf{d}$  to the relaxed LP by calling Alg. 2( $t_{wa}, r_a, p_a$ ) on non-dominated thresholds. If the expected FPR of the computed solution satisfies 5.4.1e, then we have obtained the optimal solution to LP 5.4.1 on thresholds  $\{t \mid t \geq t_{wa}\}$ . It remains to analyze the case when constraint 5.4.1e is not satisfied for  $\mathbf{d}$ .

Note that both the strategy minimizing the expected FPR analyzed in Proposition 5.4.1 and the strategy optimizing the utility for the relaxed LP 5.3.1 are produced with the same computational procedure from Alg. 2 applied on different sets of thresholds. In more details, the computation of the relaxed optimal strategy skips dominated thresholds when proceeding from the lowest threshold in the attack set to the highest one, unlike the computation of the strategy minimizing the expected FPR. Therefore, all thresholds which are neither dominated nor dominant are played with the same probability in both the strategies, see thresholds  $t_4, t_7$  in Fig. 5.4.1 for illustration. This is so because for each such a threshold, based on the algorithm, the result from the proof of Proposition 5.4.1 holds: neither probability of playing any higher-FPR thresholds can be transferred onto the threshold, nor probability of playing the threshold can be transferred onto lower-FPR thresholds without violation of 5.4.2d. Analogously, in case of the strategy  $\mathbf{d}$  for each dominant threshold it holds that no probability from lower-FPR thresholds can be transferred on the threshold without violation of 5.4.1d. Therefore, the only modifications of the  $\mathbf{d}$  which decrease the expected FPR and do not violate constraint 5.4.1e is transferring of the probability from some dominant thresholds onto their corresponding dominated thresholds. The modifications permitted with respect to 5.4.1d are illustrated with arrows in Fig. 5.4.1. Note that the transferred probability cannot be arbitrary. In order to preserve validity of 5.4.1d, the discussed restriction on maximum probability of leaving each threshold uncovered must be respected.

To sum up, the optimal solution to the FPR-restricted LP 5.4.1 can differ from the optimal solution to LP 5.3.1 in probabilities of playing some dominant and dominated thresholds. Remember we are analyzing the case when the solution to LP 5.3.1 violates 5.4.1e. Note that transferring probability from a dominant threshold onto a dominated one can be viewed as a fix of the violated constraint 5.4.1e, because the dominated threshold corresponds to lower FPR. At the same time each such fix results in increase of the defender's expected cost, it follows from the domination, see Props. 5.3.3, 5.3.7. Interestingly, the problem of finding the solution to LP 5.4.1 has a structure analogous to the well known fractional knapsack problem. Knowing that the only difference of the solution we need to find from the known solution to LP 5.3.1 can be modeled with the discussed probability transfers, we are interested in a subset of such transfers that the overall cost increase is minimized while the summarized FPR fix is higher than the minimal required one.

### Relation of the Subproblem to Fractional Knapsack

The definition of the fraction knapsack problem from ([76], p. 415) follows:

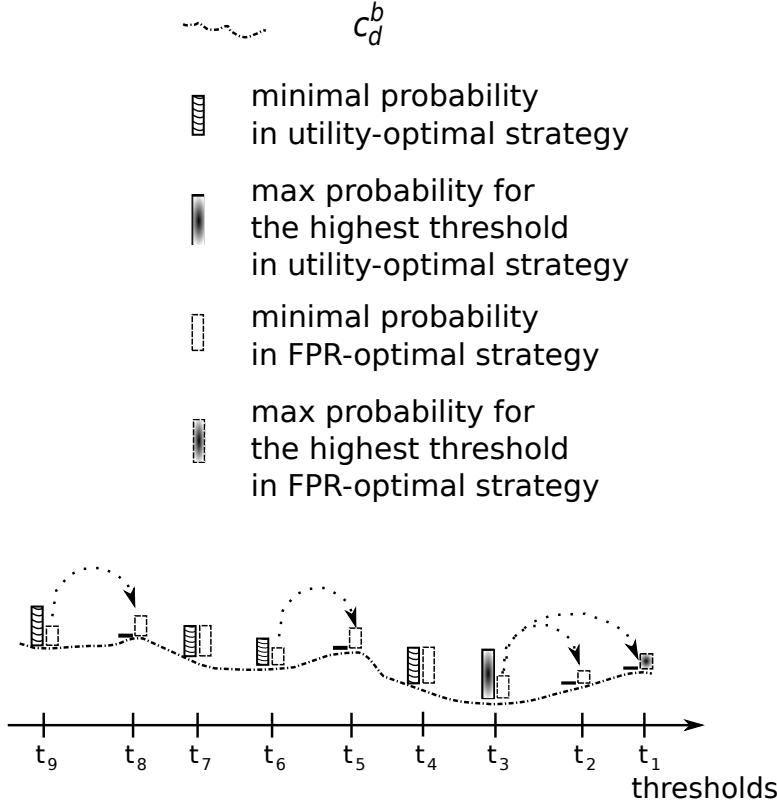


Fig. 5.4.1. Illustration of differences between the strategies

"Fractional Knapsack Problem

*Instance:* Nonnegative integers  $n, c_1, c_2, \dots, c_n, w_1, w_2, \dots, w_n, W$ .

*Task:* Find numbers  $x_1, x_2, \dots, x_n \in [0, 1]$  such that  $\sum_{j=1}^n x_j w_j < W$  and  $\sum_{j=1}^n x_j c_j$  is maximum."

In the knapsack problem,  $n$  is a number of items,  $c_i$  and  $w_i$  are the  $i^{th}$  item cost and weight correspondingly.  $W$  is the upper bound on the total weight.

In the following we draw a parallel between the fractional knapsack and the analyzed subproblem.

Let a set of dominated thresholds be denoted as  $\{t_{x1}, t_{x2}, \dots, t_{xl}\}$ . The set  $\{t_{x1}, t_{x2}, \dots, t_{xl}\}$  can be considered a set of items. Picking a fraction  $x_i$  of the  $i^{th}$  item models transferring probability  $x_i$  onto the threshold  $t_{xi}$  from its dominant threshold  $\hat{t}(t_{xi})$ .  $\hat{t}(t_{xi})$  denotes the highest threshold which dominates  $t_{xi}$ . For instance, in Fig. 5.4.1 for the threshold  $t_8$  its dominant threshold is  $t_9 = \hat{t}(t_8)$ , for the threshold  $t_5$  its dominant threshold is  $t_4 = \hat{t}(t_5)$ , and for both dominated thresholds  $t_2, t_1$  their dominant threshold is  $t_3 = \hat{t}(t_2) = \hat{t}(t_1)$ .

Item costs are to be derived based on background costs  $c_d^b$ . Note that when transferring probability  $x_i$  from the dominant threshold  $\hat{t}(t)$  onto its dominated threshold  $t$ , the defender's expected background costs are increased by  $x_i(c_d^b(t) - c_d^b(\hat{t}(t)))$ . Therefore, let the cost of the  $i^{th}$  item be  $c_i = (c_d^b(t_{xi}) - c_d^b(\hat{t}(t_{xi})))$ .

Weights will be derived based on FPR's. Note that by transferring probability  $x_i$  from a lower threshold  $t_l$  onto the higher threshold  $t_{xi}$  we reduce the expected FPR by  $x_i(\phi_{t_l} - \phi_{t_{xi}})$ . Let the weight of the  $i^{th}$  item be  $w_i = \phi_{\hat{t}(t_{xi})} - \phi_{t_{xi}}$ . Note that both  $c_i$  and  $w_i$  are non-negative real numbers.

Let the expected FPR of the utility-optimal strategy be  $\Phi$ . Then let  $W = \Phi - \Phi^{max}$ .

Let  $x_i$  denote a probability transferred onto the  $i^{th}$  threshold from its dominant threshold.

Note that even though the structure of the problem is similar to the fractional knapsack, yet we have constraint 5.4.1d, which limits maximum amount of the transferable probability on each dominated threshold. Namely, Eq. 5.3.2 gives a maximum probability  $p_{unprotected,max}(t)$  of leaving a threshold  $t$  uncovered when satisfying 5.4.1d. Let  $p_{unprotected,higher}(t)$  denote the defender's probability of playing thresholds higher than  $t$ . Let  $x_i^{UB} = p_{unprotected,max}(t_{xi}) - p_{unprotected,higher}(t_{xi})$ .

Using the introduced notation, the subproblem can be formulated as follows:

LP 5.4.1 subproblem

*Instance:* A nonnegative integer number  $n$ , non-negative real numbers  $c_1, c_2, \dots, c_n, w_1, w_2, \dots, w_n, W$ .

*Task:* Find numbers  $x_1 \in [0, x_1^{UB}], x_2 \in [0, x_2^{UB}], \dots, x_n \in [0, x_n^{UB}]$  such that  $\sum_{j=1}^n x_j w_j \geq W$  and  $\sum_{j=1}^n x_j c_j$  is minimized.

Due to the discussed similarity of the problems, a greedy algorithm for the fractional knapsack problem is adopted to solve the problem we face [77]. The adopted procedure is depicted in Alg. 4.

**Conjecture 5.4.1.** *If there exists a feasible solution to LP 5.4.1 with a threshold  $t$  being the lowest one from the attack set, then the defender's strategy produced by Alg.4 with  $t$  as an input is the optimal solution to LP 5.4.1 on thresholds not lower than  $t$ .*

The formal proof of the algorithm optimality is left for future work<sup>8</sup>. For the time being, note that we consider only thresholds corresponding to possible differences between the solution to LP 5.4.1 we seek and the solution to relaxed LP 5.3.1. In each iteration of the algorithm we pick a threshold providing a maximal decrease of the expected FPR per unit increase in background costs. Finally, once constraint 5.4.1e is satisfied we reduce the raised probability of playing a dominated threshold corresponding to the less effective decrease of the expected FPR per background costs. The probability is reduced as much as possible due to 5.4.1e.

Regarding the estimation of  $x_i^{UB}$ , a maximum probability  $p_{unprotected,max}(t_{xi})$  of leaving a threshold  $t_{xi}$  uncovered satisfying 5.4.1d is computed directly with Eq. 5.3.2. It is possible to compute the defender's probability  $p_{unprotected,higher}(t)$  of playing thresholds higher than  $t$  quite easily as well.

Initially  $p_{unprotected,higher}(t)$  of a dominated threshold is set equal to  $p_{unprotected,higher}(t_{dom})$ , where  $t_{dom}$  is the lowest dominant threshold higher than  $t$ ,  $t_{dom} = \min\{t' \mid t' \text{ non-dominant} \wedge t' > t\}$ . Based on the analyzed in detail procedure behind Alg. 2, it is exactly a probability of leaving a dominated threshold unprotected.

If  $\{t' \mid \hat{t}(t') = \hat{t}(t)\} = \{t\}$ , then  $p_{unprotected,higher}(t)$  will remain the same throughout the Alg. 4 computation. For instance, in Fig. 5.4.1 it holds for thresholds  $t_8$  and  $t_5$ .  $\hat{t}(t_8) = t_9$  does not dominate any threshold except for  $t_8$ . Thus, it is impossible that  $p_{unprotected,higher}(t)$  changes during the execution of Alg. 4. However, the threshold  $t_3$  in the figure is a dominant threshold for both  $t_2$  and  $t_1$ . Therefore, it might happen that first the probability is transferred from  $t_3$  onto  $t_1$  and later throughout the computation the probability is transferred from  $t_3$  onto  $t_2$ . The current value of  $p_{unprotected,higher}(t)$  for a dominated threshold  $t$  can be obtained when needed by means of a list  $L(t)$  of higher thresholds dominated by the same predecessor  $\hat{t}(t)$ . Once it is required to estimate  $x_i^{UB}$ ,  $p_{unprotected,higher}(t)$  can be updated as follows:  $p_{unprotected,higher}(t) =$

<sup>8</sup>An extensive empirical evaluation did not find a counterexample.

## 5. Efficient Algorithms for Optimal Solutions

**Input:**  $\phi, \hat{t}, c_d^b, W, t$ , optimal solution to LP 5.3.1 **d**.

**Output:** the defender's strategy **d**.

```

1 if the expected FPR of d satisfies 5.4.1e then
2   | return d;
3 end
4  $T'$  is a set of thresholds not lower than  $t$ ;
5  $T^d$  is a set of dominated thresholds on  $T'$ ;
6  $T_{closed}^d$  is a set of closed dominated thresholds;
   /* FPR improvement per unit background cost sacrifice */
7  $\delta(t_{xi}) = \frac{\phi_{\hat{t}(t_{xi})} - \phi_{t_{xi}}}{c_d^b(t_{xi}) - c_d^b(\hat{t}(t_{xi}))}$ ;
8  $i = 1$ ;
9  $W' = 0$ ;
10 while  $W' < W$  do
11   | Pick a threshold  $t_{xi} = \arg \min_{t \in T^d \setminus T_{closed}^d} \delta(t)$ ;
12   |  $T_{closed}^d = T_{closed}^d \cup \{t_{xi}\}$ ;
13   | Estimate  $x_i^{UB} = p_{unprotected,max}(t_{xi}) - p_{unprotected,higher}(t_{xi})$ ;
14   |  $d(t_{xi}) = x_i^{UB}$ ;
15   |  $W' = W' + x_i (\phi_{\hat{t}(t_{xi})} - \phi_{t_{xi}})$ ;
16   |  $d(\hat{t}(t_{xi})) = d(\hat{t}(t_{xi})) - x_i^{UB}$ ;
17   | Update  $\delta(t), \hat{t}$  for all  $\{t \mid t \in T^d \setminus T_{closed}^d \wedge \hat{t}(t) = \hat{t}(t_{xi})\}$ ;
18   |  $i = i + 1$ ;
19 end
20  $i = i - 1$ ;
21  $\Delta = W - \sum_{j=1}^{i-1} x_j (\phi_{\hat{t}(t_{xj})} - \phi_{t_{xj}})$ ;
22  $d(\hat{t}(t_{xi})) = d(\hat{t}(t_{xi})) + d(t_{xi}) - \frac{\Delta}{\phi_{\hat{t}(t_{xi})} - \phi_{t_{xi}}}$ ;
23  $d(t_{xi}) = \frac{\Delta}{\phi_{\hat{t}(t_{xi})} - \phi_{t_{xi}}}$ ;
24 return d;

```

**Algorithm 4:** The algorithm for solving the subproblem of LP 5.4.1.

$p_{unprotected,higher}(t) + \sum_{t_i \in L(t)} x_i$ . In practice the procedure should not be demanding. However, in the worst-case, when all thresholds are dominated by the same predecessor this can lead to quadratic time complexity of Alg. 4.

Having addressed the challenging subproblem, we summarize the whole computation of the conjectured optimal solution to LP 5.4.1 on thresholds not lower than  $t_N$  in Alg. 5.

Note that the optimal solution to LP 5.4.1 on thresholds  $\{t \mid t \geq t_N\}$  is not guaranteed to be the optimal solution to LP 5.4.1 on all thresholds in general. It is known that no threshold lower than  $t_N$  can belong to the attack set. In other words the attacker never plays any threshold lower than  $t_N$ . However, it does not imply that the defender would never play any threshold lower  $t_N$ . From Proposition 5.4.1 it is only known that the defender is unable to play such threshold as much as he might prefer to if there was no constraint on FPR. To put it more formally, it is known that there is no feasible solution with the defender playing next lower threshold  $t_{N+1}$  with maximum possible probability<sup>9</sup>. In other words, even in case there is a threshold lower than  $t_N$  which is

<sup>9</sup>Realize that in Alg. 3 the attack set expansion to lower thresholds can be viewed as follows: a

**Input:**  $\Phi^{max}$ ,  $\phi$ ,  $t_N$ ,  $r_a$ ,  $p_a$   
**Output:** the defender's strategy  $\mathbf{d}$  or no solution.

```

1 Consider only non-dominated thresholds;
2  $\mathbf{d} = \text{Alg. 3}(t_N, r_a, p_a)$ ;
3  $\Phi = \text{expected FPR of } \mathbf{d}$ ;
4 if  $\Phi \leq \Phi^{max}$  then
5 |   return  $\mathbf{d}$ ;
6 else
7 |   Consider all thresholds, including dominated;
8 |    $\mathbf{d}' = \text{Alg. 2}(t_N, r_a, p_a)$ ;
9 |    $\Phi' = \text{expected FPR of } \mathbf{d}'$ ;
10 |  while  $\Phi' > \Phi^{max}$  do
11 |    if  $t_N = t_a$  then
12 |      | return no solution;
13 |    else
14 |      | /* the next threshold higher */
15 |      |  $t_N = t_{N-1}$ ;
16 |      |  $\mathbf{d}' = \text{Alg. 2}(t_N, r_a, p_a)$ ;
17 |      |  $\Phi' = \text{expected FPR of } \mathbf{d}'$ ;
18 |    end
19 |  end
20 |  Consider only non-dominated thresholds;
21 |   $\mathbf{d} = \text{Alg. 3}(t_N, r_a, p_a)$ ;
22 |   $\Phi = \text{expected FPR of } \mathbf{d}$ ;
23 |   $\mathbf{d} = \text{Alg. 4}(\phi, \hat{t}, c_d^b, \Phi - \Phi^{max}, t_{prev}, \mathbf{d})$ ;
24 |  return  $\mathbf{d}$ ;
25 end

```

**Algorithm 5:** The algorithm for solving the LP 5.4.1.

more appealing to the defender than his current mixed strategy, the defender cannot play the optimal strategy on thresholds not lower than  $t_{N+1}$  due to 5.4.1e. However, the defender still might play the thresholds lower than  $t_N$ . Developing an algorithm for precise solution of LP 5.4.1 is left for future work. However, based on the already discovered facts we are able to estimate an upper bound on the optimal solution. There are several case which might occur. First, it might be the case that there are no thresholds lower than  $t_N$ . Then the obtained with Alg. 5 solution is optimal, given the Conjecture 5.4.1 is true. Otherwise, it is possible to upper-bound the value of the optimal solution to LP 5.4.1 with utility to the relaxed LP 5.3.1. Note that this provides a very loose upper bound. Improvement of the upper bound estimation is left for future work

We conclude the section on SSE computation under the restriction on FPR with remarks regarding the computational complexity of the devised algorithm.

Alg. 2 has linear-time complexity. On the line 15 it can be called for every threshold in the worst-case when  $t_a$  is the highest threshold, no strategy other than  $t_a$  satisfies 5.4.1e, and all thresholds belong to the relaxed solution. This results in quadratic-time complexity in the worst case. Moreover, Alg. 4 also belongs to  $\mathcal{O}(n^2)$ . Therefore, the

---

defender starts playing a lower threshold because it provides him a better outcome compared to the current mixed strategy. To optimize the utility the defender starts playing the threshold with maximum probability not violating 5.3.1d.

## 5. *Efficient Algorithms for Optimal Solutions*

developed algorithm for solving LP 5.4.1 belongs to  $\mathcal{O}(n^2)$ , where  $n$  stands for a total number of classification thresholds. As a result, overall computation of SSE under the FPR constraint using the devised algorithm is in  $\mathcal{O}(n^3)$ .



## 6. Experimental evaluation

### 6.1. Evaluation of the Algorithms Computation Efficiency

In this section we experimentally demonstrate scalability of the developed algorithms compared to the state-of-the-art general solvers. The novel algorithms outperform the solvers. In the presented set of experiments we used ROC curves of real-world intrusion detection systems (IDS) from [78]. The real-world ROC curves from [78] consist of 100 thresholds each. The scalability of the devised algorithms was also illustrated on the O2 ROC curve consisting of 201 point.

For each ROC curve several problem instances were generated with model constants initialized at random. All game constants were randomly generated from the interval  $[0; 1000]$ <sup>1</sup>. As in [36] and in the model from Chapter 4, it was assumed that the rational attacker gains from an undetected attack the same quantity the defender losses.

In the following text all confidence intervals were derived for the confidence level of 95%.

#### 6.1.1. NE computation

As a baseline for NE computation we used the Gambit [79] implementation of the algorithm computing only one NE [80]. The algorithm devised in Section 5.2 was implemented in Python without any further optimization<sup>2</sup>.

To evaluate the algorithm scalability compared to the Gambit solver, problem instances of different sizes are considered<sup>3</sup>. An instance of the problem was generated as follows: from a real-world ROC curve we picked a required number of equidistant thresholds and generated at random all constants defining the players utility functions.

First, we considered experiments on particular ROC curve with instance sizes upto maximum of 100 thresholds. For each problem size we generated 20 instances at random. The obtained results are summarized in Fig. 6.1.1.

Based on the figure, superiority of the novel algorithm is obvious.

Yet, we have also conducted a more complex set of experiments using 170 real world IDS ROC curves from [78]. We considered problems of sizes from 5 upto 80 thresholds with a step of 5. For each ROC curve and for each problem size we generate a problem instance at random. Plots summarizing all experiments for one particular ROC curve can be found on the attached CD. Based on results for all problem instances of particular size we computed an interval estimate of the mean processing time with the confidence level of 95%. The results are summarized in Fig. 6.1.2.

Due to the superiority of the developed algorithm, its processing times are impossible to read from the presented figures. The scalability of the algorithm was separately

---

<sup>1</sup>Note that the randomly generated values of the attacker's reward function were ordered, so that the assumptions of the modeling hold

<sup>2</sup>Correctness of the implementation of all algorithms was extensively verified by comparing the outputs of the implemented algorithms with the outputs of the standard solvers.

<sup>3</sup>Remember, the size of the problem instance is given by a number of considered classification thresholds.

6. Experimental evaluation

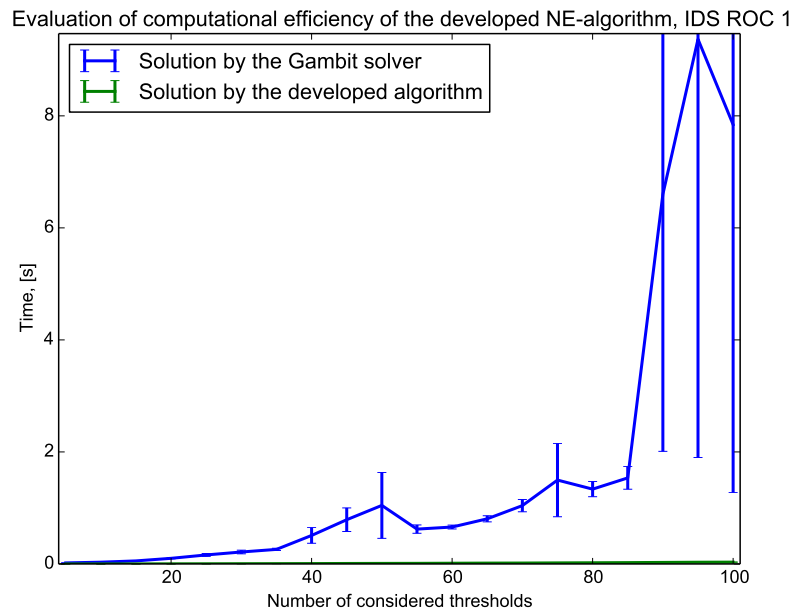


Fig. 6.1.1. NE scalability evaluation in experiments with up to 100 thresholds

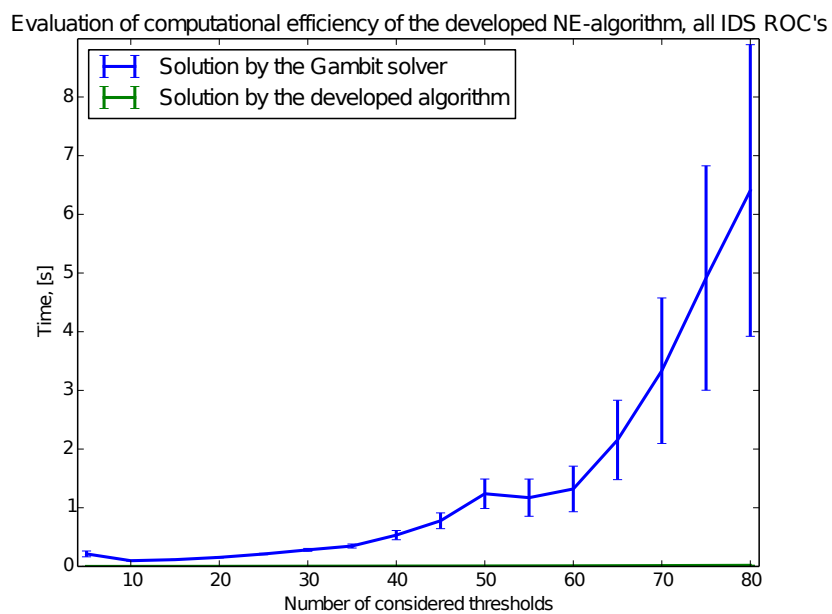
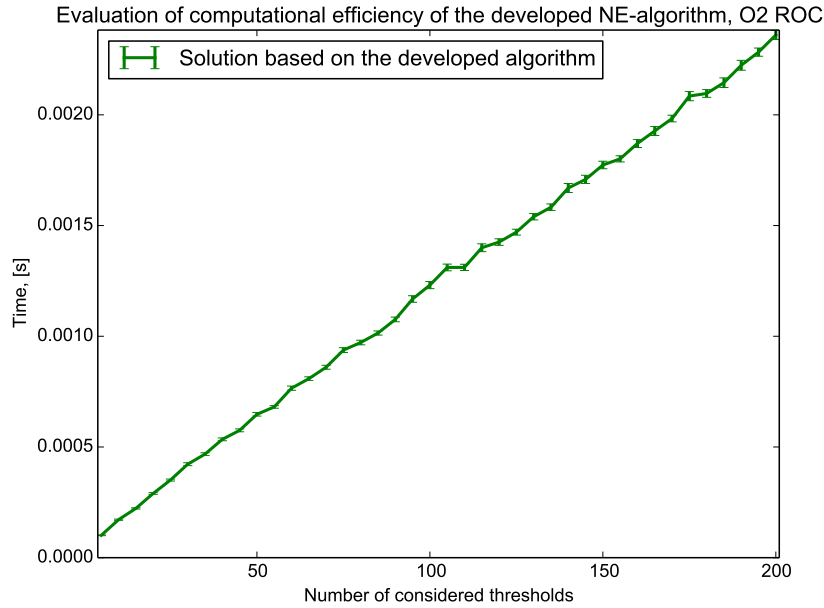


Fig. 6.1.2. NE scalability evaluation in experiments on the IDS ROC curves



**Fig. 6.1.3.** NE scalability evaluation in experiments on O2 ROC curve

verified on the O2 fraud detection ROC curve<sup>4</sup>, see Fig.6.1.3. For each number of thresholds 50 problem instances were generated at random. The results of the last experiment demonstrate the proven  $\mathcal{O}(n)$  complexity class of the algorithm. It is interesting to note slight drops in the processing time. The drops are explained with threshold domination. For each problem size we always pick the same set of equidistant thresholds. When in the picked set of thresholds there are many dominated thresholds, it effectively reduces the size of the problem from the perspective of the developed algorithm.

Also note that the processing time of the developed algorithm does not vary much. This is a further advantage of the developed procedure compared to the Gambit solver, compare Fig. 6.1.3 and Fig. 6.1.1.

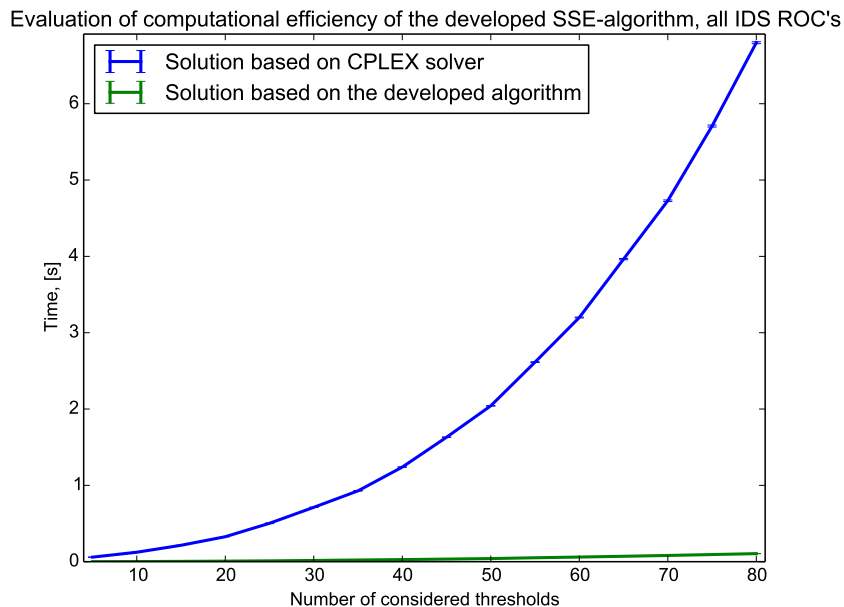
### 6.1.2. SSE computation

The method for SSE computation based on solving multiple LP with the algorithm devised in Section 5.3 was benchmarked against the method based on IBM CPLEX 12.4.

A set of all problem instances considered in the experiments was generated in the same way as in the previous section. The results of experiments for each ROC curve can be found on the attached CD. The summarization of all the results on 170 ROC curves is presented in Fig. 6.1.4.

Again, the devised algorithm significantly outperforms the standard solver. To see the scalability of the algorithm itself and its processing times, the experiments on the O2 ROC curve were performed. For each number of thresholds the results are derived based on 50 randomly generated instances. Fig. 6.1.5 depicts the results. The obtained results agree with the derived quadratic complexity of the algorithm.

<sup>4</sup>See Section 6.2.



**Fig. 6.1.4.** Scalability of the SSE computation in experiments on the IDS ROC curves

### 6.1.3. SSE computation under the restriction on FPR

Finally, we conduct analogous experiments to verify computational efficiency of the devised approximation algorithm for computation of SSE solution under the restriction on expected FPR. Furthermore, we experimentally estimate how far the computed solution might be from the optimal one. The set of conducted experiments to obtain the results regarding the FPR-restricted SSE computation was designed in the same way as experiments for evaluation of the NE and SSE algorithms. Results of all experiments for individual ROC curves can be found on the attached CD.

The results of the scalability experiments are presented in Fig. 6.1.6.

From Fig. 6.1.6, summarizing all the experiments on the real world IDS ROC curves, it follows that the developed algorithm is computationally more efficient compared to the IBM CPLEX benchmark.

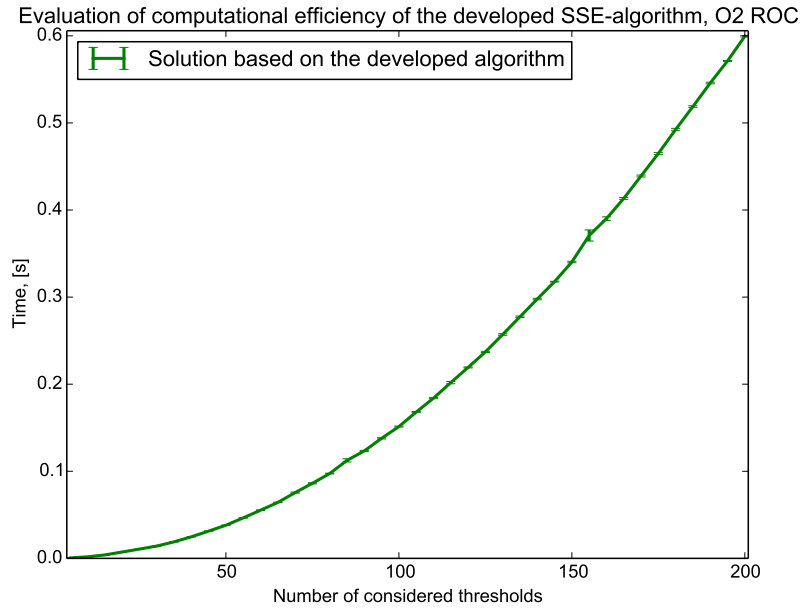
Furthermore, we tested scalability of the algorithm on the O2 ROC curve. a PyPy optimization of our algorithm implementation. The results of the scalability test are depicted in Fig. 6.1.7.

Notice the drop in computation time which occurred around the problem size of 100 thresholds. After the investigation it turned out that in the problem instances generated on 105 equidistant thresholds the number of subproblems without solution decreased significantly. It might be interesting to investigate the issue in more details in future work.

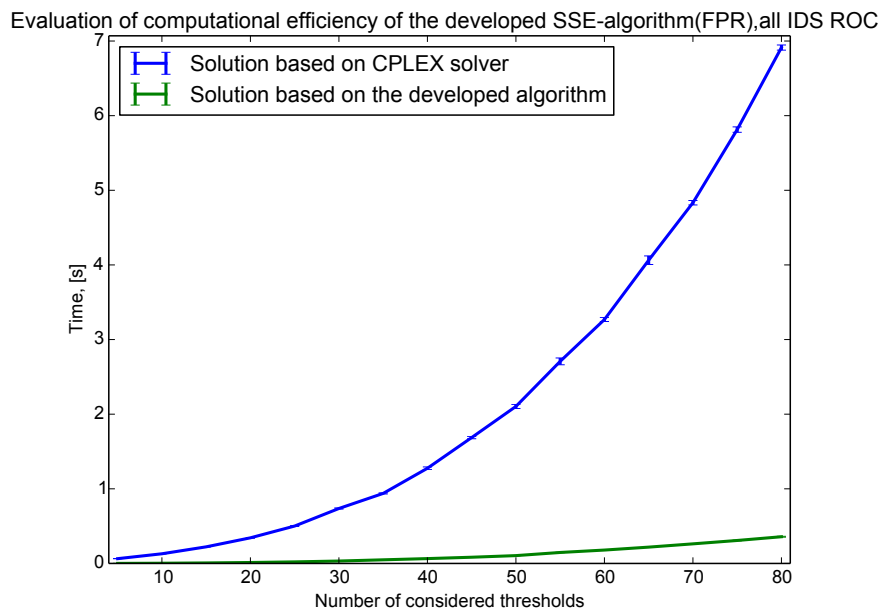
Next, we experimentally evaluated the quality of the solution returned with the devised algorithm. The utilities from different experiments were normalized with the value of the corresponding optimal utility<sup>5</sup>. For illustrative purposes we present costs of the strategy, which are the negated utility. The results are summarized in Fig. 6.1.8.

Notice that the restriction on FPR influences the resulting utility significantly. In

<sup>5</sup>Without the normalization it would be impossible to compare the results of different experiments, as all instances were generated in random which resulted in different scales in utility functions.

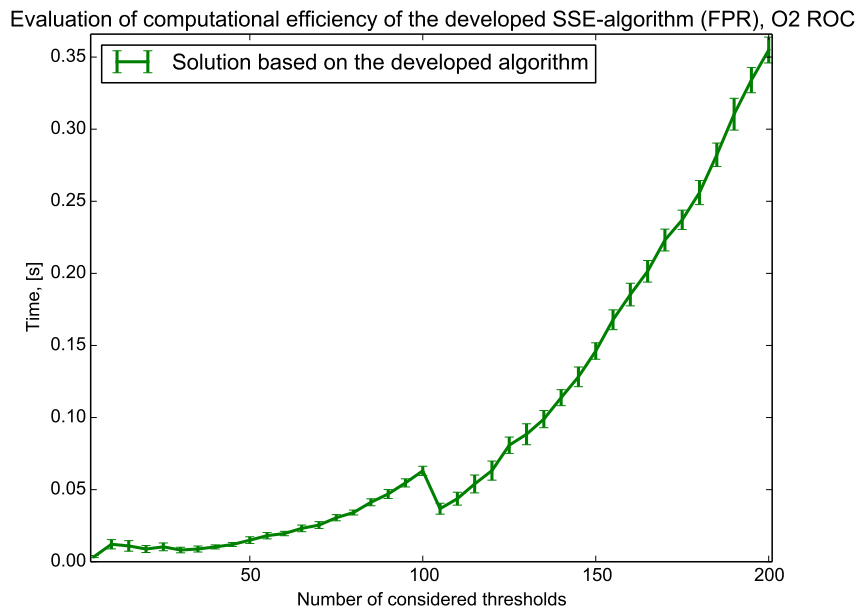


**Fig. 6.1.5.** Scalability of the SSE computation in experiments on the O2 ROC curve

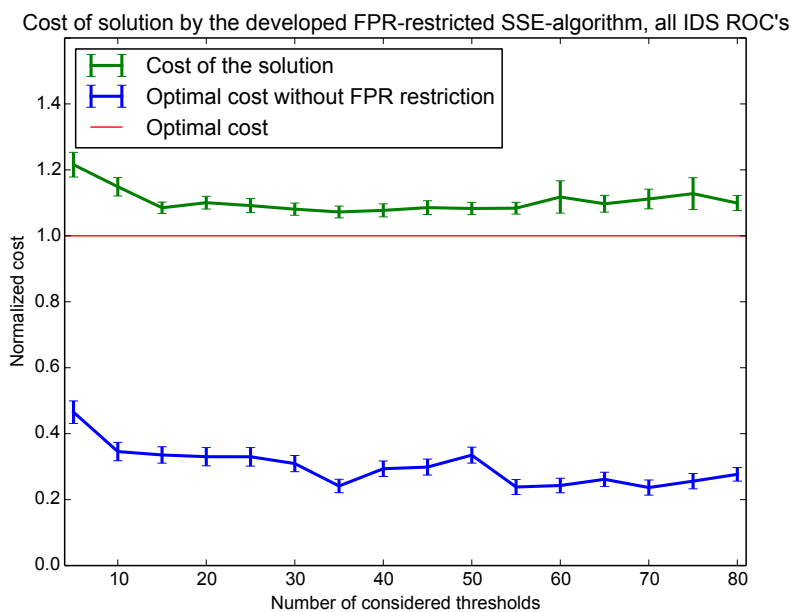


**Fig. 6.1.6.** Scalability of the SSE computation under the restriction on FPR, IDS ROC curves

## 6. Experimental evaluation



**Fig. 6.1.7.** Scalability of the SSE computation, FPR restriction



**Fig. 6.1.8.** Suboptimality of the SSE computation under the FPR restriction

order to satisfy the FPR constraint, a considerable amount of the utility must be sacrificed. Note that the decrease in utility is comparable for both the optimal solution and the solution by the devised algorithm. Yet, computationally the devised algorithm is significantly superior.

## 6.2. Evaluation of Developed Adversarial Classification on the O2 CZ application

In this section we demonstrate applicability of the developed method for solving a real world adversarial classification problem: the O2 CZ fraud detection case.

Evaluation is based on a standard procedure for machine learning classifier performance evaluation: out of sample error is estimated using testing dataset, and game theoretic modeling is based on validation dataset.

All collected data (20 484 samples, out of which 1941 are positive) were divided into 3 datasets:

- a) a training dataset (containing approximately 60% of all collected data: 12257 samples, out of which 1162 are positive),
- b) a validation dataset (approximately 20% of all data: 4112 samples, out of which 382 are positive),
- c) a test dataset (approximately 20% of all data: 4115 samples, out of which 397 are positive).

Note that the stratified random split of data into the datasets was performed based on an international mobile subscriber identity (IMSI), which is unique. Thus, in case there were samples regarding the same IMSI, all of them would go to the same dataset.

Unlike classical machine learning, we are interested not only the out-of-sample static error, but also it is required to address robustness against potential adaptive adversaries. Thus, besides evaluation of the resulting adversary-aware classifier on the training dataset, we also evaluate the performance on altered test dataset. Several altered datasets were produced for the purpose of extensive evaluation. The attacker altered the dataset first based on his NE optimal strategy, next based on his best response to the defender's SSE strategy, then based on the FPR-SSE optimal strategy and finally based on the best response to the defender's optimal fixed threshold. Moreover, for each altering scenario a separate altered testing set was produced for different probability  $P_a$  of an arbitrary attacker to be adaptive. Finally, we address robustness of the classification against active adversaries, which do not use strategic reasoning. In this final scenario the testing dataset is altered as follows: an attacker modifies his feature vector to a feature vector of the attribute-nearest user. By the attribute-nearest neighbour we mean a user with smallest euclidean distance in the feature space. Again, the modification of the training dataset was performed for different values of  $P_a$ .

We consider the developed FPR-restricting algorithm for SSE computation the most advanced from the applicable point of view. We benchmark the algorithm against several standard classification methods:

- a) classify as fraudulent all instances with probability of being malicious exceeding 50% (termed a max-probability method in the following text);
- b) using the validation dataset estimate a classification threshold corresponding to the maximum acceptable FPR and use it for classification (for the sake of conciseness, in the following the method is termed FPR-threshold).

Furthermore, we benchmark the algorithm against the developed NE and SSE based algorithms, as well as against the adversary-aware fixed optimal threshold, termed in

the following text an adversary-optimal threshold.

As performance criteria we use FPR, TPR, and total financial losses due to the undetected attackers. The latter are estimated from the data the way it was discussed in Section 4.1.4.

All interval estimates of the criteria values derived based on the repeated experiments were computed for the confidence level of 95%.

Lastly, we are about to comment the FPR acceptable in the case of O2 CZ fraud detection. The module, which is currently being deployed, is intended to serve as the last line of defense<sup>6</sup>. Based on the data from previous year, it is expected that the module would process approximately 50 legitimate users daily. 1 FP per day was decided to be reasonable. As a result, acceptable FPR was set to 2%.

### 6.2.1. Utilities Derived based on the O2 CZ Data

Before diving into the verification, we comment on the model utilities derived based on the validation dataset. Due to the advantageous data-driven approach developed, there is no need to hypothesize regarding the utility functions. The only thing which is needed to be estimated based on the domain knowledge is the expected level of attacker adaptivity, controlled with the model parameters  $\gamma$  and  $P_a$ .

After the classifier<sup>7</sup> was created using the training dataset, the validation dataset was used to produce an ROC curve, see Fig. 6.2.1.

The R package [68] was used to obtain and manipulate an ROC curve. The R script with functions used for the game theoretic model estimation can be found on the attached CD.

The obtained ROC curve was used to estimate FPR corresponding to different classification thresholds. Initially we picked 400 classification thresholds as follows: we considered thresholding probabilities ranging from 0.0025 upto 0.9975 with a step of 0.0025. For each considered value we picked an ROC threshold which was the lowest one exceeding the value.

Next, the reward of a rational attacker was estimated based on Eqs. 5.1.5, 4.1.3. As for real-world decision boundaries there is likely no efficient procedure to optimize  $l_d^{FN}(x)$ , we assumed that the detected attacker would imitate the closest undetected samples from the whole collected dataset. The attacker reward function estimated based on Eqs. 5.1.5, 4.1.3, 4.1.1 is depicted in Fig. 6.2.2.

The depicted reward function was generated with probability of the attacker's adaptivity  $P_a = 30\%$ . Note that  $P_a$  does not change the shape of the attacker's reward. Its sole purpose is to enable management to set the expected trade-off between background costs and costs due to rational attackers. The chosen value of parameter was  $\gamma = 2$ . The higher the  $\gamma$ , the quicker the attacker's reward increases in thresholds<sup>8</sup>.

Note an interesting observation regarding the  $\gamma$  parameter: the lower the threshold is, the more sensitive to  $\gamma$  the attacker's reward is. See Fig. 6.2.3.

To put it differently, the attacker's natural type tends to correspond to attacks on high thresholds. This empirically supports the modeling assumption from [36] adapted

<sup>6</sup>See Section 2.2.1 for more details on the machine learning module developed for the O2 CZ fraud detection division. Here we just remind that the module was deployed as the last line of defense, facing a challenging classification task of detecting fraudulent customers which a previous version of the complex system would fail to catch.

<sup>7</sup>Even though the underlying machine learning classifier is not a subject of the thesis, in Section 2.2.1 the reader can find more information and interesting links regarding the fraud detection.

<sup>8</sup>Once more we remind that the numbering of the thresholds starts from the highest threshold. In such setting on an ROC curve the first threshold corresponds to zero on the x-axis.



### ROC curve of the O2 CZ classifier

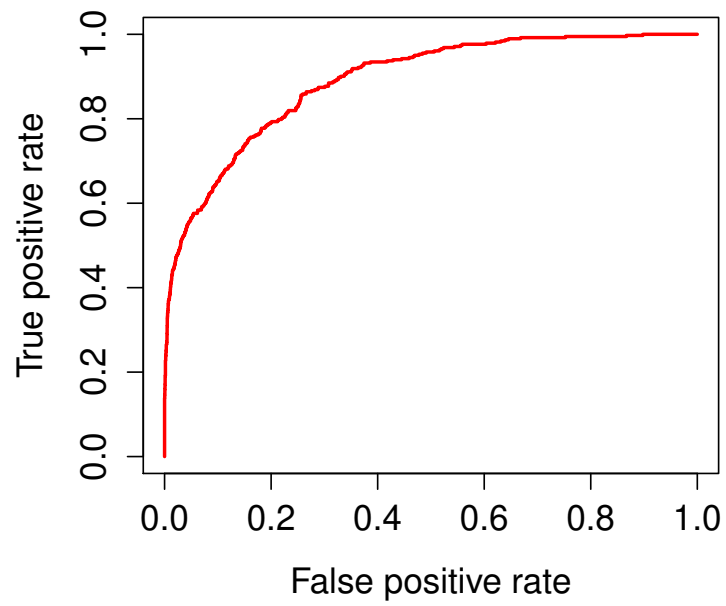


Fig. 6.2.1. ROC curve of the O2 CZ fraud detector

### The attacker's reward function

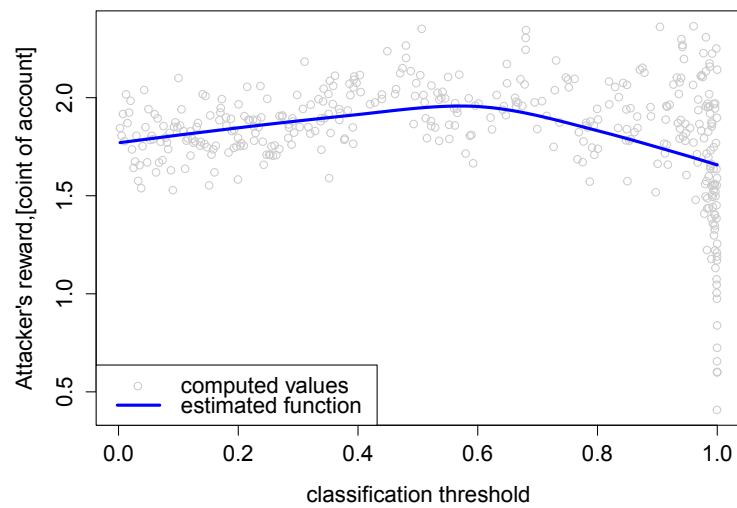


Fig. 6.2.2. The function of the attacker's reward  $r_a$

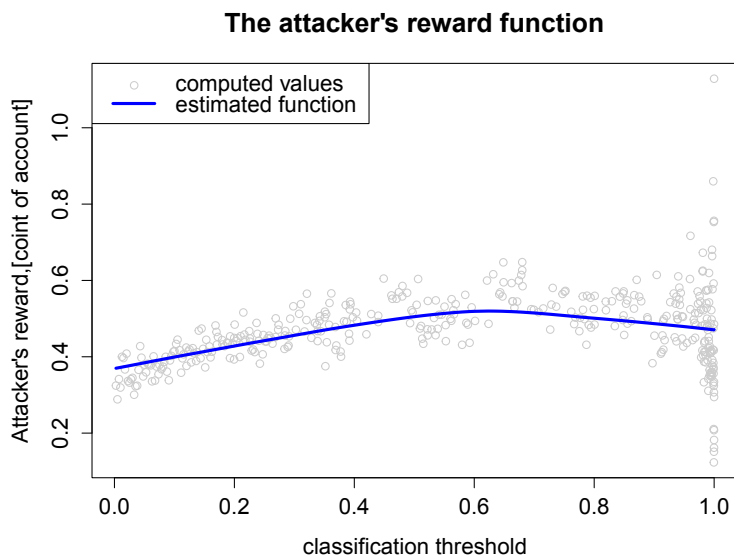


Fig. 6.2.3. The function of the attacker's reward  $r_a$  for  $\gamma = 20$

in this work.

Notice that the estimated reward function is not non-decreasing, contradicting the assumption of the model. However, the rational attacker would never play thresholds from a decreasing part of the reward function, as for each such threshold there is another threshold corresponding to higher reward with lower protection. We filter out thresholds which can be never attacked. Rigorous analysis of the case of arbitrary reward functions is left for the future work. Note that another fix to the occurred situation might be setting a high value for  $\gamma$ .

Next, the set of FP's  $X^{FP}$  and stationary FN's  $X_s^{FN}$  were obtained for every classification threshold from the obtained set. A cost of processing a single FP was estimated based on internal expenses for the operation of the fraud detection department. The resulting function of the defender's costs due to stationary population<sup>9</sup> is depicted in Fig. 6.2.4. Note that in order to prevent revealing of the company's sensitive data, all money-based attributes were rescaled to the range chosen at random. All money-based features were rescaled to the same range. A unit of the range is to be called a coin of account.

In order to reduce the effect of noise, based on the obtained cost values the final functions were derived using the R function `loess` for local polynomial regression fitting.

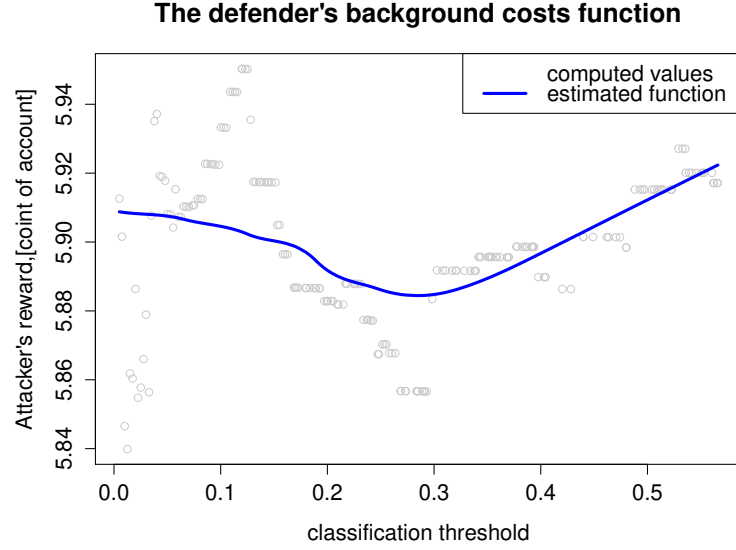
## 6.2.2. Performance

### Out-of-sample Performance Estimation

The results of the performance evaluation of all the classification methods on the test dataset are summarized in Table 6.2.2. To evaluate the randomized strategies 1000 repeated experiments were performed.

From the values in the table we can see that even though both the FPR-threshold and FPR-SSE randomized strategy does not exceed the maximum expected FPR of 2% on

<sup>9</sup>See Eqs. 5.1.3,4.1.2.



**Fig. 6.2.4.** The function of the defender's background costs  $c_d^b$

|                                 | Classification method |               |                             |              |              |              |
|---------------------------------|-----------------------|---------------|-----------------------------|--------------|--------------|--------------|
|                                 | Max-probability       | FPR-threshold | adversary-optimal threshold | NE           | SSE          | FPR-SSE      |
| FPR,%                           | 1.3%                  | 2.04          | 17.3                        | 2.75 ± 0.00  | 2.75 ± 0.00  | 2.09 ± 0.00  |
| TPR,%                           | 41.08                 | 45.00         | 76.74                       | 48.04 ± 0.03 | 48.04 ± 0.02 | 44.76 ± 0.03 |
| Financial loss, coin of account | 694.0                 | 670.5         | 372.2                       | 622.6 ± 0.2  | 622.6 ± 0.2  | 672.6 ± 0.4  |

**Tab. 6.2.1.** Out-of-sample performance estimated on the test dataset

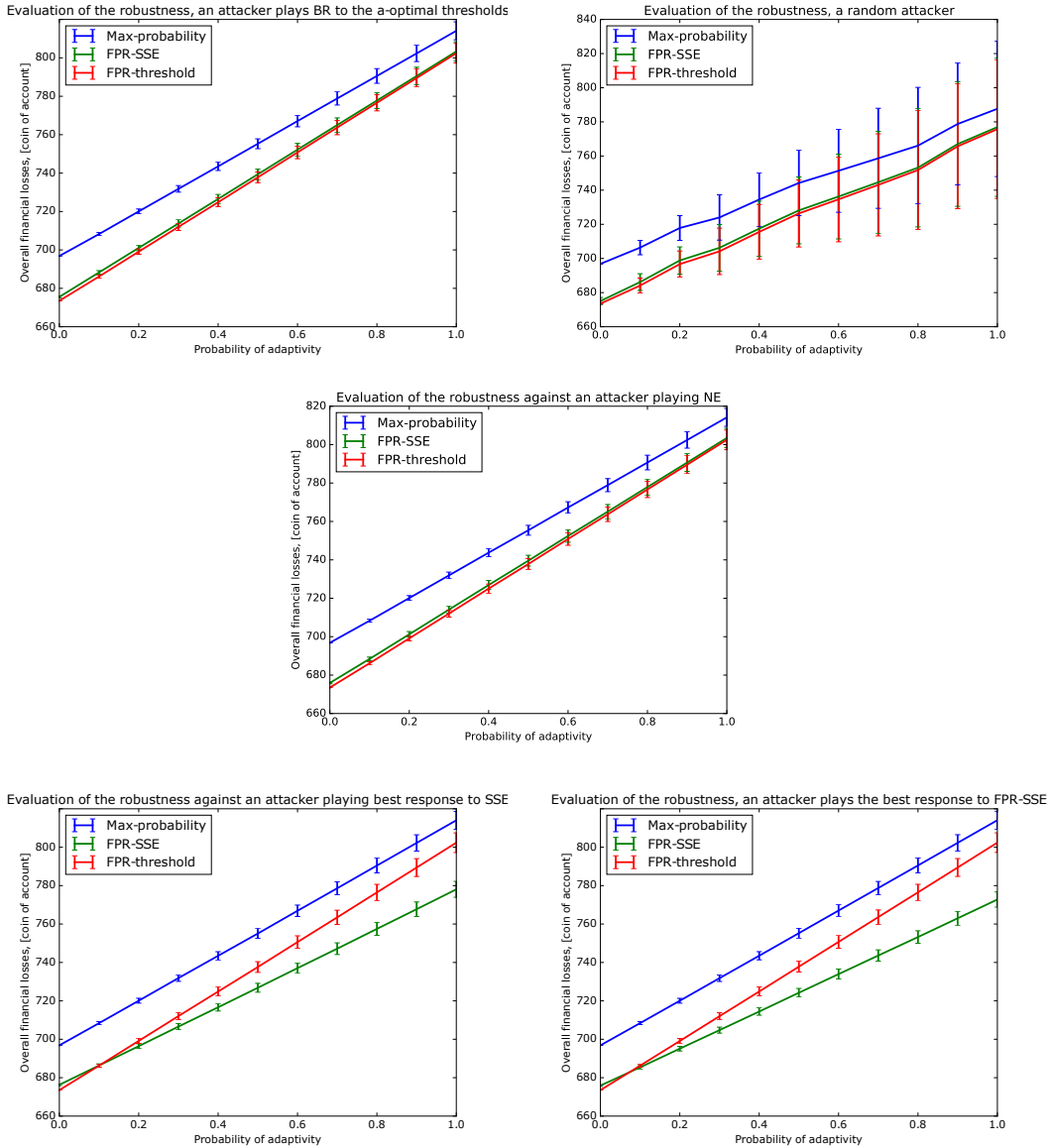
the validation dataset, on the test dataset both of them slightly exceeded the restriction. However, in case of the developed module for the fraud detection system 0.1% increase in FPR can be tolerated. The adversary-optimal threshold does not take FPR into consideration. As a result, the deterministic strategy violates the FPR constraint by a great margin. Neither NE nor SSE considers the FPR constraint. However, in case of the particular application, by chance, both strategies coincide and violate the constraint by 0.75%, which is significantly less compared to the adversary-optimal threshold. The standard max-probability classification does not violate the FPR constraint, however it corresponds to the lowest TPR out of all options. From the table it follows that without violating the FPR constraint the optimal TPR is approximately 45%. It is important to note that the devised FPR restricted SSE strategy provides on the static data performance analogous to the classification threshold derived based on ROC curve. To sum up, enriching the model with adversary-aware modeling did not result in decrease of the classifier performance. Thus, the developed method satisfies the goal of remaining

## 6. Experimental evaluation

highly accurate.

### Performance Estimation on the Altered Datasets

We consider the strategies with the FPR out-of-sample estimate roughly satisfying the restriction on FPR: FPR-threshold, FPR-SSE, Max-probability.



**Fig. 6.2.5.** Results of robustness evaluations

The results of the robustness evaluation experiments are depicted in Fig. 6.2.5. From the figure one can see that in several experiments costs of both FPR-threshold and FPR-SSE strategies increase in the same way with increasing probability of adversarial adaptability. Max-probability strategy expectedly has higher costs due to smaller TPR. An interesting result can be seen in the case the attacker plays the strategy a defender assumes him to. In this case the robustness of classification optimized with the devised game theoretic approach is better compared to a non-optimized version.

## 7. Conclusion

The research presented in this thesis was set out to explore the possibility of developing a method applicable in practice and optimizing robustness of machine learning classifiers against rational entities. After analysis of different types of attacks against machine learning classifiers in Chapter 2, we focused on the most common type: adaptive attempts to avoid detection by the deployed classifier. As it was stated in Chapter 1, adaptivity of rational entities is a known and serious problem, causing the industry significant financial losses. Yet, classical machine learning methods are not capable to take the problem into consideration as they were designed based on the assumption that future observations would follow the same distribution the training data do. Thus, the only remaining option for classical machine learning methods is to neglect the dangerous problem and deal with it and its consequences once the problem occurs. As discussed in Section 2.3, one common way to deal with the arisen problem is to retrain the classifier. However, this does not limit capabilities of the adversary and he will be able to adapt again, harming the system to the same extent as before. Moreover, as discussed in Section 2.1, instant retraining of the classifier might make the system vulnerable to another type of attacks: poisoning of the classifier. To sum up, the main problem with this approach is that the security designer passively allows an adversary to repeatedly harm the system as much as the adversary wishes, while the security designer just humbly repairs the damage in a reactive manner and starts a new cycle of the never ending process. A better approach is to somehow limit the ability of the adversary to cause harm. As it was discussed in Section 1.1, in both theoretical and empirical studies it was found that randomization of a classifier decision boundary mitigates the capability of the adversary to adapt. Furthermore, from the discussions presented in Sections 2.3 and 1.1 it follows that rather than just blindly add a random noise to the decision boundary, a more promising approach is to understand intentions and capabilities of the attacker, to understand the way the adversary reacts to the defender's actions and based on the findings derive a randomization optimal against the understood adversary. In Chapter 1 game theory was pointed out as an efficient tool for understanding an adversary and modeling interactions with him. After the required background on game theory was refreshed in Section 3.1, we analyzed the existing methods of game theoretic optimization of detecting malicious behavior in Section 3.3. Yet, we found out that there is a gap between existing methods and practical requirements of the industry. Understanding of realistic practical requirements on the approach was enabled by the author's work on a machine learning module for an internal fraud detection system of O2 Czech Republic telecommunications company, the module was introduced in Section 2.2.1. Besides increasing the classifier robustness against attackers, further crucial requirements, discussed in Section 1.2, were:

*a)* The method must preserve performance of the already deployed classifier. Thus, unlike several existing approaches, the novel approach must not put any restrictions on the learning algorithm and is required to be applied to any state-of-the-art method as to a black box.

*b)* The approach is not allowed to lead to unreasonably high false positive rate, as in real-world security classification the majority of the classified instances are legitimate

## 7. Conclusion

and high rate of false alarms would result in the denial of service.

*c)* The underlying model must be realistic, with a minimum number of hypotheses unfounded with observations. It forces the model to be data-driven.

*d)* It is required to address the fact that there are very different types of adversaries. Taking this fact into consideration, the model must provide a general security against the whole population of attackers.

*e)* It is required to create a model which does not assume that all adversaries are rational and adaptive. The idea is to formulate a model which can limit damages by an expected fraction of adaptive adversaries, while classifying correctly the stationary population.

*f)* Optimization of the machine learning classifier based on the model must be computationally efficient, so that it would be possible to compute the solution in practice. Moreover, it should be possible to compute the solution in the future once more data are gathered.

Capitalizing on the most advanced related work identified in Section 3.3, in Chapter 4 we developed a model addressing the requirements *a)*, *c)* - *e)*. The presented method of game theoretic optimization does not put any restrictions on an underlying machine learning classifier. The only requirement on the classifier is to produce a quantitative estimate representing the degree to which an instance is adversarial. If this holds, it is possible to derive an ROC curve and formulate the model. Estimation of utilities in the model is fully based on data. The only domain knowledge which is required to produce the model is an estimation of adversarial adaptability. Furthermore, the cost for false alarms also must be derived based on internal expenses for operation of a security division.

Once the promising game theoretic model was formulated, it was necessary to develop efficient algorithms for computing solutions in the model, including a solution with a control over false positive rate. Deep analysis of the formulated problem and discovering several interesting facts about it in Chapter 5 enabled us to come up with efficient algorithms for computing several game theoretic solution concepts. Besides a classical concept of Nash equilibrium and a concept of Strong Stackelberg equilibrium, popular in security applications of game theory, we tackled the newly presented concept of Strong Stackelberg equilibrium under the restriction on false positive rate. For both NE and SSE we managed to devise algorithms for precise computation of the solutions. For SSE under FPR restriction we came up with an approximation algorithm. Even though the computation of NE is in general a hard problem belonging to PPAD complexity class, the presented algorithm computing NE in the model has linear worst-case complexity. The algorithm for computation of SSE in the model has quadratic worst-case complexity and the algorithm for FPR restricted SSE belongs to  $\mathcal{O}(n^3)$ . All the algorithms were benchmarked against existing alternatives in a set of extensive experiments on real-world ROC curves of intrusion detection systems in Chapter 6. The experiments showed an impressive scalability of the developed algorithms. Finally, we conduct another set of experiments demonstrating applicability of the proposed model for solving real world problems. In Section 6.2 it is demonstrated how the model can be derived in practical applications of adversarial classification. Next, the strategy addressing FPR is computed based on the developed model and then it is compared to several baselines: strategies based on NE and SSE concept, which ignore FPR, an optimal fixed threshold based on the developed game theoretic model, a strategy of classifying as malicious any sample with probability of being malicious exceeding 50%, and a fixed threshold based on allowed FPR. The results of the evaluation show that the proposed game theoretic optimization did not result in a notable decrease of performance compared to

other methods respecting FPR constraint, while the proposed approach can improve robustness of the method against rational adversaries.

To sum up, we managed to address all points from the thesis assignment: the existing models are analyzed in Chapter 3, a novel data-driven model is formulated in Chapter 4, several efficient algorithms are designed in Chapter 5, and finally evaluations of the results efficiency and their practical applicability are presented in Chapter 6.

Even though we managed to solve several important problems, there is a lot of work left for future research. For instance, probably the most exciting extension of our model for the future work might be developing a data-driven approach based on clustering of the observed malicious instances and defining attacker types using the clusters. Perhaps, Gaussian Mixture Model (GGM) clustering might be a promising method of choice. Using the derived from collected data GGM it would be possible to estimate for each new instance its probability of being malicious and the cluster-based type of a probable attacker. Using this idea it might be possible to derive a type-specific security. In other words a defender would use different classification strategies for different types of the attacker. It would be required to develop a method for estimation of the players' utility functions from the suggested GGM-based attacker types.

Another open problem is NE computation under the restriction on FPR. We share our ideas regarding it in Appendix C.

# Appendix A.

## Abbreviations

|       |                                   |
|-------|-----------------------------------|
| LP    | Linear Programming                |
| O2 CZ | O2 Czech Republic, a.s.           |
| NE    | Nash Equilibrium                  |
| SE    | Stackelberg Equilibrium           |
| SSE   | Strong Stackelberg Equilibrium    |
| ROC   | Receiver Operating Characteristic |
| TP    | True Positive                     |
| FN    | False Negative                    |
| TN    | True Negative                     |
| FP    | False Positive                    |
| TPR   | True Positive Rate                |
| FNR   | False Negative Rate               |
| TNR   | True Negative Rate                |
| FPR   | False Positive Rate               |
| AUC   | Area under ROC curve              |



# Appendix B.

## Model Notation

|                             |   |
|-----------------------------|---|
| $\mathbf{T}$                | a set of classification thresholds  |
| $t_d$                       | a classification threshold chosen by the defender   |
| $t_a$                       | a classification threshold guessed by the attacker  |
| $X$                         | a set of malicious feature vectors in the dataset, or a set of all attacker types                 |
| $l_d^{FN}(x)$               | the defender's loss when failed to detect a malicious instance with a feature vector $x$          |
| $c_{FP}$                    | the defender's cost for processing a single FP  |
| $X_s^{FN}$                  | a set of undetected attacker types with a stationary attacker                                     |
| $X_a^{FN}$                  | a set of types undetected due to adaptability of the rational attacker                            |
| $X^{FP}$                    | a set of the produced FP's  |
| $P_a$                       | probability that the attacker is rational and adaptive  |
| $c_d^b$                     | the defender's background costs due to stationary population                                      |
| $c_d^r$                     | the defender's costs due to undetected attack by a rational attacker                              |
| $r_a$                       | the attacker's reward function for undetected attack  |
| $U_d^u$                     | the defender's utility for playing a fixed threshold  |
| $\Phi^{max}$                | maximum tolerable expected FPR  |
| $\phi_t$                    | FPR corresponding to the fixed classification threshold $t$                                       |
| $p_{unprotected,max}(t)$    | maximum probability of leaving a threshold $t$ uncovered satisfying 5.3.1d, 5.4.1d, see Eq. 5.3.2 |
| $p_{unprotected,higher}(t)$ | the defender's total probability of playing thresholds higher than $t$                            |

## Appendix C.

# Ideas regarding computation of NE under the restriction on FPR

We conclude this chapter with a few remarks regarding the NE under the restriction on FPR, which is left for future work. We identify a subroutine which would likely be part of the algorithm computing the constrained NE and then prove that the subroutine can be done efficiently.

In Proposition 5.2.2 it was proven that the highest threshold from the set of non-dominated thresholds  $\mathbf{T}_r$  always belongs to the support of the defender's NE strategy. Let denote this threshold  $t' = \max\{\mathbf{T}_r\}$ . It should be possible to prove that for several special cases  $t'$  belongs to the support of the defender's strategy under the FPR-restriction as well. The main difficulty in the future work proofs would likely arise due to the fact that the discovered action domination would hold only partially if there is the restriction on FPR. For instance, consider two thresholds  $t_2$  and  $t_1$ ,  $t_2 < t_1$ , such that both thresholds are non-dominated when there is no restriction on FPR. In case neither  $t_1$  nor  $t_2$  corresponds to FPR higher than  $\Phi^{max}$ , the domination analogous to the one in this work obviously holds. However, it is required to analyze in future work what would happen if, for instance,  $\phi_{t_1} \leq \Phi^{max} \wedge \phi_{t_2} > \Phi^{max}$ .

For the case when the highest non-dominated threshold  $t'$  does not exceed the limit  $\Phi^{max}$  and therefore belongs to the NE support, an iterative procedure analogous to Alg. 1 would likely be a part of the NE solution computation.

Note that the threshold  $t'$  corresponds to the lowest FPR out of all non-dominated thresholds  $\mathbf{T}_r$ <sup>1</sup>. If the highest threshold  $t'$  satisfies  $\phi_{t'} \leq \Phi^{max}$ , then the defender would play  $t'$  and might also consider playing lower thresholds. In Prop. 5.2.3 the way of the defender's NE support computation was discovered based on the fact that until there is a lower threshold more appealing to the defender than  $t'$ , it contradicts Prop. 5.2.2. Using this finding, in Alg. 1 the attacker expended the support of the defender's NE strategy until there were lower thresholds not included in the current estimation of the support and until the highest non-included threshold had a background cost lower than the defender's current expected cost for playing  $t'$ , see Fig. 5.2.3. To put it differently, in case of the addressed in this work unconstrained problem there are two reasons why the defender would not consider playing any lower thresholds instead of his current support estimate: either there are no lower thresholds or their background costs are higher than the expected utility for playing the current support. Computation of NE strategy in Alg. 1 starts from thresholds corresponding to lower FPR and proceeds to thresholds with higher FPR. It is likely that in order to enforce the FPR restriction, one more terminating condition would be introduced into an analogous algorithm: the

---

<sup>1</sup>Remember, a set of thresholds  $\mathbf{T}$  was defined based on the set of FPR's  $\mathbf{F}$  obtained from the real-world ROC curve as follows: for every unique TPR of the generated ROC curve only the minimal FPR is considered. However, nothing changes if thresholds other than  $\mathbf{T}$  are taken into consideration as well. In this case FPR is non-increasing in thresholds.

defender would not consider a lower threshold if playing it would lead to expected FPR exceeding  $\Phi^{max}$ .

In order to introduce the third terminating condition into Alg. 1, we must be able to determine the defender's expected FPR for a current support. A straightforward way is to call Alg. 2 for each support, compute the defender's strategy and check whether or not  $\sum_{t=t_1}^{t_n} d_t \phi_t \leq \Phi^{max}$ . However, Alg. 2 has a linear time complexity, therefore the resulting approach would have quadratic time complexity in the worst-case. It turns out that we can do better than that. After adding a threshold to the support it is possible to compute the expected FPR of a new support without computation of the defender's strategy. The procedure is similar to estimation of the defender's expected cost in Alg. 3 for SSE computation.

**Proposition C.0.1.** *Consider any iteration of the loop in Alg. 1. Note that each iteration can be identified by the value of  $N$  which is equal to a number of thresholds in the current support. If in the considered iteration of the algorithm the defender's expected FPR is equal to  $\Phi$  and the algorithm execution continues to the next iteration, then in the next iteration of the algorithm the expected FPR would be equal to*

$$\Phi' = \frac{r_a(t_N) - r_a(t_{N+1})}{r_a(t_N) + p_a} \phi_{t_{N+1}} + \frac{r_a(t_{N+1}) + p_a}{r_a(t_N) + p_a} \Phi.$$

**Proof:** The proposition is to be proven by mathematical induction.

*Base case*

As a base case consider the first iteration, when  $N = 1$ , in other words  $t_N = t'$ .

At the beginning of the computation in Alg. 1 the defender's support consists only of the highest reasonable threshold  $t_N = t'$ . A current expected FPR  $\Phi$  is directly  $\Phi = \phi_{t'}$ . After adding the next lower threshold  $t_{N+1}$ , the defender's NE support would be  $\{t_{N+1}, t_N\}$ . Computation of the defender's mixed NE strategy under the assumption that his strategy support is  $\{t_{N+1}, t_N\}$  can be done using Alg. 2. Remember, the rational attacker considers only  $\{t_{N+1}, t_N\}$  in this case, and the main idea behind Alg. 2 is to make the attacker indifferent between all thresholds in the support, i.e. between  $t_{N+1}$  and  $t_N$ . In order to do so the defender must play  $t_{N+1}$  with probability  $d(t_{N+1})$  satisfying

$$r_a(t_{N+1}) = (1 - d(t_{N+1})) \cdot r_a(t_N) - p_a \cdot d(t_{N+1}) \implies d(t_{N+1}) = \frac{r_a(t_N) - r_a(t_{N+1})}{r_a(t_N) + p_a}.$$

Therefore, in case the next threshold would be added to the support, the new defender's expected FRP would be

$$\Phi' = d(t_{N+1}) \phi_{t_{N+1}} + (1 - d(t_{N+1})) \Phi = \frac{r_a(t_N) - r_a(t_{N+1})}{r_a(t_N) + p_a} \phi_{t_{N+1}} + \frac{r_a(t_{N+1}) + p_a}{r_a(t_N) + p_a} \Phi.$$

*Inductive step*

Let us consider an iteration of the algorithm with  $N = n$ . The inductive hypothesis states that the gradually computed expected FPR is correct, i.e.,  $\Phi = \sum_{i=1}^n d(t_i) \phi_{t_i}$ , where  $\mathbf{d} = (d(t_n), \dots, d(t_1))$  stands for the defender's NE mixed strategy if the NE support is  $\{t_n, \dots, t_1\}$ . To put the hypothesis differently,  $\mathbf{d}$  is the result of Alg. 2 with  $N = n$  as an input. Thus,

$$\forall i \in \{2, 3, \dots, n\} : d(t_i) = \frac{r_a(t_n) + p_a}{r_a(t_i) + p_a} - \frac{r_a(t_n) + p_a}{r_a(t_{i-1}) + p_a} =$$

$$= (r_a(t_n) + p_a) \left( \frac{1}{r_a(t_i) + p_a} - \frac{1}{r_a(t_{i-1}) + p_a} \right),$$

and for  $t_1 : d(t_1) = \frac{r_a(t_n) + p_a}{r_a(t_1) + p_a}$ .

Let us now compute the value  $\Phi'$  of the next iteration of the algorithm with  $N' = N + 1 = n + 1$ . For this iteration the correct value of the expected FPR is  $\Phi' = \sum_{i=1}^{n+1} d'(t_i) \phi_{t_i}$ , where  $\mathbf{d}' = (d'(t_{n+1}), \dots, d'(t_1))$  is the defender's NE mixed strategy obtained by means of Alg. 2 with  $N' = n + 1$  as an input. In other words,

$$\forall i \in \{2, 3, \dots, n + 1\} : d'(t_i) = (r_a(t_{n+1}) + p_a) \left( \frac{1}{r_a(t_i) + p_a} - \frac{1}{r_a(t_{i-1}) + p_a} \right),$$

and for  $t_1 : d'(t_1) = \frac{r_a(t_{n+1}) + p_a}{r_a(t_1) + p_a}$ .

Note,

$$\begin{aligned} d'(t_{n+1}) = (r_a(t_{n+1}) + p_a) \left( \frac{1}{r_a(t_{n+1}) + p_a} - \frac{1}{r_a(t_n) + p_a} \right) &\iff d'(t_{n+1}) = 1 - \frac{r_a(t_{n+1}) + p_a}{r_a(t_n) + p_a} \iff \\ &\iff r_a(t_{n+1}) + p_a = (1 - d'(t_{n+1}))(r_a(t_n) + p_a). \end{aligned}$$

Therefore,

$$\begin{aligned} \forall i \in \{2, 3, \dots, n\} : d'(t_i) &= (r_a(t_{n+1}) + p_a) \left( \frac{1}{r_a(t_i) + p_a} - \frac{1}{r_a(t_{i-1}) + p_a} \right) = \\ &= (1 - d'(t_{n+1}))(r_a(t_n) + p_a) \left( \frac{1}{r_a(t_i) + p_a} - \frac{1}{r_a(t_{i-1}) + p_a} \right) = (1 - d'(t_{n+1}))d(t_i), \end{aligned}$$

and analogously  $d'(t_1) = (1 - d'(t_{n+1}))d(t_1)$ .

Hence, the correct value of the expected FPR in the next iteration can be rewritten as

$$\begin{aligned} \Phi' &= \sum_{i=1}^{n+1} d'(t_i) \phi_{t_i} = d'(t_{n+1}) \phi_{t_{n+1}} + \sum_{i=1}^n d'(t_i) \phi_{t_i} = d'(t_{n+1}) \phi_{t_{n+1}} + (1 - d'(t_{n+1})) \sum_{i=1}^n d(t_i) \phi_{t_i} = \\ &= d'(t_{n+1}) \phi_{t_{n+1}} + (1 - d'(t_{n+1})) \Phi \xrightarrow{d'(t_{n+1}) = 1 - \frac{r_a(t_{n+1}) + p_a}{r_a(t_n) + p_a}} \\ &\implies \Phi' = \frac{r_a(t_n) - r_a(t_{n+1})}{r_a(t_n) + p_a} \phi_{t_{n+1}} + \frac{r_a(t_{n+1}) + p_a}{r_a(t_n) + p_a} \Phi \iff \\ &\iff \Phi' = \frac{r_a(t_N) - r_a(t_{N+1})}{r_a(t_N) + p_a} \phi_{t_{N+1}} + \frac{r_a(t_{N+1}) + p_a}{r_a(t_N) + p_a} \Phi. \blacksquare \end{aligned}$$

Further analysis of NE computation under the restriction on FPR is not within the scope of this thesis and is left for future work.

## Appendix D.

### The attached CD contents

- An implementation of the developed algorithms. In addition, input files with the used O2 ROC curve, the defender's background costs and the attacker's rewards are provided. All files and source codes are located in a directory *scripts/algorithms*. To compute the randomized strategies launch `main.py`. Source codes of the algorithms can be found in a file `implemented_algorithms.py`. Moreover, an R script `manipulating_data.R` used for manipulation with the dataset, derivation of the utilities and conducting robustness experiments is provided as well.
- Results of numerical experiments carried out on IDS ROC curves in the archive `Results of experiments on IDS ROC curves.zip`. Names of the subfolders are self-explanatory.
- Electronic version of the thesis in PDF.

## References

- [1] Jessica Fridrich. *Steganography in digital media: principles, algorithms, and applications*. Cambridge University Press, 2009.
- [2] Giorgio Giacinto, Fabio Roli, and Luca Didaci. “Fusion of multiple classifiers for intrusion detection in computer networks”. In: *Pattern recognition letters* 24.12 (2003), pp. 1795–1803.
- [3] Mehran Sahami et al. “A Bayesian approach to filtering junk e-mail”. In: *Learning for Text Categorization: Papers from the 1998 workshop*. Vol. 62. 1998, pp. 98–105.
- [4] Salvatore J Stolfo et al. “Detecting viral propagations using email behavior profiles”. In: *ACM Transactions on Internet Technology (TOIT)* (2004), pp. 128–132.
- [5] Robin Sommer and Vern Paxson. “Outside the closed world: On using machine learning for network intrusion detection”. In: *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE. 2010, pp. 305–316.
- [6] Arun A Ross, Karthik Nandakumar, and Anil Jain. *Handbook of multibiometrics*. Vol. 6. Springer Science & Business Media, 2006.
- [7] Ingemar Cox et al. *Digital watermarking and steganography*. Morgan Kaufmann, 2007.
- [8] Clifton Phua et al. “A comprehensive survey of data mining-based fraud detection research”. In: *arXiv preprint arXiv:1009.6119* (2010).
- [9] Francisca Nonyelum Ogwueleka. “Data mining application in credit card fraud detection system”. In: *Journal of Engineering Science and Technology* 6.3 (2011), pp. 311–322.
- [10] Yeshinegus Getaneh. “Predictive Modeling for Fraud Detection In Telecommunications”. PhD thesis. AAU, 2013.
- [11] Tom Fawcett and Foster Provost. “Adaptive fraud detection”. In: *Data mining and knowledge discovery* 1.3 (1997), pp. 291–316.
- [12] Ulrich Flegel, Julien Vayssière, and Gunter Bitz. “A state of the art survey of fraud detection technology”. In: *Insider threats in cyber security*. Springer, 2010, pp. 73–84.
- [13] Aisha Abdallah, Mohd Aizaini Maarof, and Anazida Zainal. “Fraud Detection System: A survey”. In: *Journal of Network and Computer Applications* (2016).
- [14] Constantinos S Hilas. “Data mining approaches to fraud detection in telecommunications”. In: *2nd Pan-Hellenic Conference on Electronics and Telecommunications PACET*. Vol. 12. 2012.
- [15] Hamid Farvaresh and Mohammad Mehdi Sepehri. “A data mining framework for detecting subscription fraud in telecommunication”. In: *Engineering Applications of Artificial Intelligence* 24.1 (2011), pp. 182–194.

- [16] Peter Burge et al. “Fraud detection and management in mobile telecommunications networks”. In: *Security and Detection, 1997. ECOS 97., European Conference on*. IET. 1997, pp. 91–96.
- [17] Michiaki Taniguchi et al. “Fraud detection in communication networks using neural and probabilistic methods”. In: *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*. Vol. 2. IEEE. 1998, pp. 1241–1244.
- [18] Liyiming Ke, Bo Li, and Yevgeniy Vorobeychik. “Behavioral Experiments in Email Filter Evasion”. In: (2016).
- [19] Yevgeniy Vorobeychik and Bo Li. “Optimal randomized classification in adversarial settings”. In: *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems. 2014, pp. 485–492.
- [20] Battista Biggio, Giorgio Fumera, and Fabio Roli. “Security evaluation of pattern classifiers under attack”. In: *Knowledge and Data Engineering, IEEE Transactions on* 26.4 (2014), pp. 984–996.
- [21] Richard Colbaugh and Kristin Glass. “Predictive defense against evolving adversaries”. In: *Intelligence and Security Informatics (ISI), 2012 IEEE International Conference on*. IEEE. 2012, pp. 18–23.
- [22] Battista Biggio et al. “Evasion attacks against machine learning at test time”. In: *Machine Learning and Knowledge Discovery in Databases*. Springer, 2013, pp. 387–402.
- [23] Battista Biggio, Giorgio Fumera, and Fabio Roli. “Adversarial pattern classification using multiple classifiers and randomisation”. In: *Structural, Syntactic, and Statistical Pattern Recognition*. Springer, 2008, pp. 500–509.
- [24] Nilesh Dalvi et al. “Adversarial classification”. In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2004, pp. 99–108.
- [25] Murat Kantarcioglu, Bowei Xi, and Chris Clifton. “A Game Theoretical Framework for Adversarial Learning”. In: *CERIAS 9th Annual Information Security Symposium*. Citeseer. 2008.
- [26] Ling Huang et al. “Adversarial machine learning”. In: *Proceedings of the 4th ACM workshop on Security and artificial intelligence*. ACM. 2011, pp. 43–58.
- [27] Tom Fawcett. “In vivo spam filtering: a challenge problem for KDD”. In: *ACM SIGKDD Explorations Newsletter* 5.2 (2003), pp. 140–148.
- [28] Yufeng Kou et al. “Survey of fraud detection techniques”. In: *Networking, sensing and control, 2004 IEEE international conference on*. Vol. 2. IEEE. 2004, pp. 749–754.
- [29] Gang Wang et al. “Man vs. machine: Practical adversarial detection of malicious crowdsourcing workers”. In: *23rd USENIX Security Symposium (USENIX Security 14)*. 2014, pp. 239–254.
- [30] Battista Biggio, Giorgio Fumera, and Fabio Roli. “Multiple classifier systems for robust classifier design in adversarial environments”. In: *International Journal of Machine Learning and Cybernetics* 1.1-4 (2010), pp. 27–41.

- [31] Nir Kshetri. *The global cybercrime industry: economic, institutional and strategic perspectives*. Springer Science & Business Media, 2010.
- [32] Phil Gosset and Mark Hyland. “Classification, detection and prosecution of fraud in mobile networks”. In: *Proceedings of ACTS mobile summit, Sorrento, Italy* (1999).
- [33] The Federal Communications Commission. *Cell Phone Fraud*. 2016. URL: <https://www.fcc.gov/consumers/guides/cell-phone-fraud> (visited on 04/10/2016).
- [34] Aron Laszka et al. “Optimal thresholds for intrusion detection systems”. In: *Proceedings of the Symposium and Bootcamp on the Science of Security*. ACM. 2016, pp. 72–81.
- [35] Yevgeniy Vorobeychik and John Ross Wallrabenstein. “Using machine learning for operational decisions in adversarial environments”. In: *International Joint Workshop on Optimisation in Multi-Agent Systems and Distributed Constraint Reasoning (OptMAS-DCR)*. 2014.
- [36] Viliam Lisý, Robert Kessl, and Tomáš Pevný. “Randomized Operating Point Selection in Adversarial Classification”. In: *Machine Learning and Knowledge Discovery in Databases*. Springer, 2014, pp. 240–255.
- [37] Mengchen Zhao, Bo An, and Christopher Kiekintveld. “Optimizing Personalized Email Filtering Thresholds to Mitigate Sequential Spear Phishing Attacks”. In: (2016).
- [38] Tansu Alpcan and Tamer Başar. “A game theoretic analysis of intrusion detection in access control systems”. In: *Decision and Control, 2004. CDC. 43rd IEEE Conference on*. Vol. 2. IEEE. 2004, pp. 1568–1573.
- [39] Tansu Alpcan and Tamer Basar. “A game theoretic approach to decision and analysis in network intrusion detection”. In: *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*. Vol. 3. IEEE. 2003, pp. 2595–2600.
- [40] Michael Brückner and Tobias Scheffer. “Nash equilibria of static prediction games”. In: *Advances in neural information processing systems*. 2009, pp. 171–179.
- [41] Milind Tambe. *Security and game theory: Algorithms, deployed systems, lessons learned*. Cambridge University Press, 2011.
- [42] Manish Jain et al. “Software assistants for randomized patrol planning for the lax airport police and the federal air marshal service”. In: *Interfaces* 40.4 (2010), pp. 267–290.
- [43] Christopher Kiekintveld et al. “Computing optimal randomized resource allocations for massive security games”. In: *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems. 2009, pp. 689–696.
- [44] Praveen Paruchuri et al. “Playing games for security: An efficient exact algorithm for solving Bayesian Stackelberg games”. In: *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2*. International Foundation for Autonomous Agents and Multiagent Systems. 2008, pp. 895–902.
- [45] Marco Barreno et al. “Can machine learning be secure?” In: *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*. ACM. 2006, pp. 16–25.



- [46] LEMONIA Dritisoula, Patrick LOISEAU, and John MUSACCHIO. “Computing the nash equilibria of intruder classification games”. In: *Decision and Game Theory for Security*. Springer, 2012, pp. 78–97.
- [47] ARON LASZKA, Yevgeniy VOROBAYCHIK, and Xenofon D Koutsoukos. “Optimal Personalized Filtering Against Spear-Phishing Attacks.” In: *AAAI*. 2015, pp. 958–964.
- [48] Mengchen Zhao, Bo An, and Christopher Kiekintveld. “An Initial Study on Personalized Filtering Thresholds in Defending Sequential Spear Phishing Attacks”. In: *Proceedings of the 2015 IJCAI Workshop on Behavioral, Economic and Computational Intelligence for Security*. 2015.
- [49] Kim Zetter. *Researchers uncover RSA phishing attack, hiding in plain sight*. 2011.
- [50] Bo Li and Yevgeniy Vorobeychik. “Scalable Optimization of Randomized Operational Decisions in Adversarial Classification Settings.” In: *AISTATS*. 2015.
- [51] Michail I Schlesinger and Václav Hlaváč. *Ten lectures on statistical and structural pattern recognition*. Vol. 24. Springer Science & Business Media, 2013.
- [52] Foster Provost and Tom Fawcett. “Robust classification for imprecise environments”. In: *Machine learning* 42.3 (2001), pp. 203–231.
- [53] Roger B Myerson. *Game theory*. Harvard university press, 2013.
- [54] Aron Laszka, Jian Lou, and Yevgeniy Vorobeychik. “Multi-Defender Strategic Filtering Against Spear-Phishing Attacks”. In: (2016).
- [55] Wei Liu and Sanjay Chawla. “A game theoretical model for adversarial learning”. In: *Data Mining Workshops, 2009. ICDMW’09. IEEE International Conference on*. IEEE. 2009, pp. 25–30.
- [56] Pavel Laskov and Marius Kloft. “A framework for quantitative security analysis of machine learning”. In: *Proceedings of the 2nd ACM workshop on Security and artificial intelligence*. ACM. 2009, pp. 1–4.
- [57] Murat Kantarcioğlu, Bowei Xi, and Chris Clifton. “Classifier evaluation and attribute selection against active adversaries”. In: *Data Mining and Knowledge Discovery* 22.1-2 (2011), pp. 291–335.
- [58] Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification*. John Wiley & Sons, 2012.
- [59] the UK’s national reporting centre for fraud Action Fraud and cyber crime. *Telecommunications frauds*. 2016. URL: <http://www.actionfraud.police.uk/fraud-az-telecommunications-fraud> (visited on 04/10/2016).
- [60] the UK’s national reporting centre for fraud Action Fraud and cyber crime. *Premium rate phone line scams*. 2016. URL: <http://www.actionfraud.police.uk/fraud-az-premium-rate-phone-line-scams> (visited on 04/10/2016).
- [61] Josef Kittler et al. “On combining classifiers”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 20.3 (1998), pp. 226–239.
- [62] Tianqi Chen and Carlos Guestrin. “XGBoost: A Scalable Tree Boosting System”. In: *arXiv preprint arXiv:1603.02754* (2016).
- [63] Evgenia Dimitriadou et al. “Package ‘e1071’”. In: *R Software package, available at <http://cran.rproject.org/web/packages/e1071/index.html>* (2009).

- [64] Kymie MC Tan, Kevin S Killourhy, and Roy A Maxion. “Undermining an anomaly-based intrusion detection system using common exploits”. In: *Recent Advances in Intrusion Detection*. Springer. 2002, pp. 54–73.
- [65] Y. Shoham and K. Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2009. URL: <http://www.masfoundations.org/mas.pdf> (visited on 07/05/2013).
- [66] Zhengyu Yin et al. “Stackelberg vs. Nash in security games: Interchangeability, equivalence, and uniqueness”. In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems. 2010, pp. 1139–1146.
- [67] Tom Fawcett. “An introduction to ROC analysis”. In: *Pattern recognition letters* 27.8 (2006), pp. 861–874.
- [68] Tobias Sing et al. “ROCR: visualizing classifier performance in R”. In: *Bioinformatics* 21.20 (2005), pp. 3940–3941.
- [69] Peng Liu, Wanyu Zang, and Meng Yu. “Incentive-based modeling and inference of attacker intent, objectives, and strategies”. In: *ACM Transactions on Information and System Security (TISSEC)* 8.1 (2005), pp. 78–118.
- [70] Kong-wei Lye and Jeannette M Wing. “Game strategies in network security”. In: *International Journal of Information Security* 4.1 (2005), pp. 71–86.
- [71] Quanyan Zhu and Tamer Başar. “Indices of power in optimal IDS default configuration: theory and examples”. In: *Decision and Game Theory for Security*. Springer, 2011, pp. 7–21.
- [72] Michael Brückner and Tobias Scheffer. “Stackelberg games for adversarial prediction problems”. In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2011, pp. 547–555.
- [73] Alireza Naeimi Sadigh, Sattar Hashemi, and Ali Hamzeh. “Spam detection by Stackelberg game”. In: *Advanced Computing: An International Journal* 2.2 (2011), pp. 32–40.
- [74] Constantinos Daskalakis, Paul W Goldberg, and Christos H Papadimitriou. “The complexity of computing a Nash equilibrium”. In: *SIAM Journal on Computing* 39.1 (2009), pp. 195–259.
- [75] Vincent Conitzer and Tuomas Sandholm. “Computing the optimal strategy to commit to”. In: *Proceedings of the 7th ACM conference on Electronic commerce*. ACM. 2006, pp. 82–90.
- [76] H Bernhard, BH Korte, and J Vygen. *Combinatorial optimization: Theory and algorithms*. 2008.
- [77] George B Dantzig. “Discrete-variable extremum problems”. In: *Operations research* 5.2 (1957), pp. 266–288.
- [78] Tomas Pevny, Markus Rehak, and Martin Grill. “Detecting anomalous network hosts by means of pca”. In: *Information Forensics and Security (WIFS), 2012 IEEE International Workshop on*. IEEE. 2012, pp. 103–108.
- [79] Richard D McKelvey, Andrew M McLennan, and Theodore L Turocy. *Gambit: Software tools for game theory, version 14.1. 0*. 2014.

- [80] Carlton E Lemke and Joseph T Howson Jr. “Equilibrium points of bimatrix games”. In: *Journal of the Society for Industrial and Applied Mathematics* 12.2 (1964), pp. 413–423.