Journal of Machine Learning Research 13 (2012) 3681-3724 Submitted 2/10; Revised 2/11; Published 11/12

Security Analysis of Online Centroid Anomaly Detection

Marius Kloft[*] KLOFT@TU-BERLIN.DE
Machine Learing Group
Technische Universität Berlin
Franklinstr. 28/29
10587 Berlin, Germany

Pavel Laskov PAVEL.LASKOV@UNI-TUEBINGEN.DE
Wilhelm-Schickard Institute for Computer Science
University of Tübingen
Sand 1
72076 Tübingen, Germany

Editor: Ingo Steinwart

Abstract

Security issues are crucial in a number of machine learning applications, especially in scenarios dealing with human activity rather than natural phenomena (e.g., information ranking, spam detection, malware detection, etc.). In such cases, learning algorithms may have to cope with manipulated data aimed at hampering decision making. Although some previous work addressed the issue of handling malicious data in the context of supervised learning, very little is known about the behavior of anomaly detection methods in such scenarios. In this contribution,[1] we analyze the performance of a particular method—online centroid anomaly detection—in the presence of adversarial noise. Our analysis addresses the following security-related issues: formalization of learning and attack processes, derivation of an optimal attack, and analysis of attack efficiency and limitations. We derive bounds on the effectiveness of a poisoning attack against centroid anomaly detection under different conditions: attacker's full or limited control over the traffic and bounded false positive rate. Our bounds show that whereas a poisoning attack can be effectively staged in the unconstrained case, it can be made arbitrarily difficult (a strict upper bound on the attacker's gain) if external constraints are properly used. Our experimental evaluation, carried out on real traces of HTTP and exploit traffic, confirms the tightness of our theoretical bounds and the practicality of our protection mechanisms.

Keywords: anomaly detection, adversarial, security analysis, support vector data description, computer security, network intrusion detection

1. Introduction

Machine learning methods have been instrumental in enabling novel data analysis applications. Numerous currently indispensable technologies—object recognition, user preference analysis, spam filtering, to name only a few—rely on accurate analysis of massive amounts of data. Unfortunately, the increasing *use* of machine learning methods gives rise to a threat of their *abuse*. A convinc-

∗. MK is also with Department of Brain and Cognitive Engineering, Korea University, Anam-dong, Seongbuk-gu, Seoul 136-713, Korea. Parts of the work were done while MK was with Computer Science Division and Department of Statistics, University of California, Berkeley, CA 94720-1758, USA.

1. A short version of this paper appeared previously in AISTATS 2010.

ing example of this phenomenon are emails that bypass spam protection tools. Abuse of machine learning can take on various forms. A malicious party may affect the training data, for example, when it is gathered from the real operation of a system and cannot be manually verified. Another possibility is to manipulate objects observed by a deployed learning system (test data) so as to bias its decisions in favor of an attacker. Yet another way to defeat a learning system is to send a large amount of nonsense data in order to produce an unacceptable number of false alarms and hence force the system's operator to turn it off. Manipulation of a learning system may thus range from simple cheating to complete disruption of its operation.

A potential insecurity of machine learning methods stems from the fact that they are usually not designed with adversarial input in mind. Starting from the mainstream computational learning theory (Vapnik, 1998; Schölkopf and Smola, 2002), a prevalent assumption is that training and test data are generated from the same fixed, but unknown, probability distribution. This assumption obviously does not hold for adversarial scenarios. Furthermore, even the recent work on learning with non-i.i.d. data (Steinwart et al., 2009; Mohri and Rostamizadeh, 2010) or differing training and test distributions (Sugiyama et al., 2007) is not necessarily appropriate for adversarial input, because in the latter case one must account for a specific worst-case difference while all the aforementioned papers assume that the data is generated stochastically.

Computer security is the most important application field in which robustness of learning algorithms against adversarial input is crucial. Modern security infrastructures are facing an increasing professionalization of attacks motivated by monetary profit. A widespread deployment of evasion techniques, such as encryption, obfuscation and polymorphism, is manifested in a rapidly increasing diversity of malicious software observed by security experts. Machine learning methods offer a powerful tool to counter a rapid evolution of security threats. For example, anomaly detection can identify unusual events that potentially contain novel, previously unseen exploits (Wang and Stolfo, 2004; Rieck and Laskov, 2006; Wang et al., 2006; Rieck and Laskov, 2007). Another typical application of learning methods is automatic signature generation which drastically reduces the time needed for development and deployment of attack signatures (Newsome et al., 2006; Li et al., 2006). Machine learning methods can also help researchers better understand the design of malicious software by using classification or clustering techniques together with special malware acquisition and monitoring tools (Bailey et al., 2007; Rieck et al., 2008).

In order for machine learning methods to be successful in security applications—and in general in any application where adversarial input may be encountered—they should be equipped with countermeasures against potential attacks. The current understanding of security properties of learning algorithms is rather incomplete. Earlier work in the PAC-framework addressed some scenarios in which training data is deliberately corrupted (Angluin and Laird, 1988; Littlestone, 1988; Kearns and Li, 1993; Auer, 1997; Bschouty et al., 1999). These results, however, are not connected to modern learning algorithms used in classification, regression and anomaly detection problems. Several examples of effective attacks were demonstrated in the context of specific security and spam detection applications (Lowd and Meek, 2005a; Fogla et al., 2006; Fogla and Lee, 2006; Perdisci et al., 2006; Newsome et al., 2006; Nelson et al., 2008), which motivated a recent work on taxonomization of such attacks (Barreno et al., 2006, 2008, 2010). However, it remains largely unclear whether machine learning methods can be protected against adversarial impact.

We believe that an unequivocal answer to the problem of the "security of machine learning" does not exist. Security guarantees cannot be established experimentally, because the notion of security addresses events that do not just happen on average but rather only potentially may happen. Hence,

a theoretical analysis of machine learning algorithms for adversarial scenarios is indispensable. It is hard to imagine, however, that such an analysis can offer meaningful results for any attack in every circumstance. Hence, to be a useful guide for practical applications of machine learning in adversarial environments, such an analysis must address *specific attacks against specific learning algorithms*. This is precisely the approach followed in this contribution.

The main focus of our work is the security analysis of online centroid anomaly detection against the so-called "poisoning" attacks. Centroid anomaly detection is a very simple method which has been widely used in computer security applications (e.g., Forrest et al., 1996; Warrender et al., 1999; Wang and Stolfo, 2004; Rieck and Laskov, 2006; Wang et al., 2006; Rieck and Laskov, 2007). In the learning phase, centroid anomaly detection computes the mean of all training data points:

$$\mathbf{c} = \frac{1}{n} \sum_{j=1}^{n} \mathbf{x}_j \,.$$

Detection is carried out by computing the distance of a new example $\mathbf{x}$ from the centroid $\mathbf{c}$ and comparing it with an appropriate threshold:

$$f(\mathbf{x}) = \begin{cases} 1, & \text{if } ||\mathbf{x} - \mathbf{c}|| > \theta \\ 0, & \text{otherwise} \,. \end{cases}$$

Notice that all operations can be carried out using kernel functions—a standard trick known since the development of support vector machines and kernel PCA (Boser et al., 1992; Schölkopf et al., 1998; Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004), which substantially increases the discriminative power of this method.

More often than not, anomaly detection algorithms are deployed in non-stationary environments and need to be regularly re-trained. Since the data is fed into the learning phase without any verification, an adversary has an opportunity to force a learning algorithm to learn a representation suitable for him. One particular kind of attack is the so-called "poisoning" in which specially crafted data points are injected to cause the decision function to misclassify a given malicious point as benign. This attack makes sense when an attacker does not have "write" permission to the training data, hence cannot manipulate it directly. Therefore, his goal is to trick an algorithm by merely using an "append" permission, by sending new data that looks innocuous to the learning algorithm but changes the algorithm's state in a way that favors the attacker; for example, forcing the algorithm to accept a specific attack point later during the testing stage.

The poisoning attack against online centroid anomaly detection has been considered by Nelson and Joseph (2006) for the case of an infinite window, that is, when a learning algorithm memorizes all data seen so far. Their main result was surprisingly optimistic: it was shown that the number of attack data points that must be injected grows exponentially as a function of the impact over a learned hypothesis. However, the assumption of an infinite window also hinders the ability of a learning algorithm to adjust to legitimate changes in the data distribution.

1.1 Contributions of This Work

As a main contribution, we present the security analysis of online centroid anomaly detection for the *finite window* case; that is, when only a fixed number of data points can be used at any time to form a hypothesis. We show that, in this case, an attacker can easily compromise a learning algorithm by

using only a linear amount of injected data unless additional constraints are imposed. As a further contribution, we analyze the algorithm under two additional constraints: (a) the fraction of the traffic controlled by an attacker is bounded by ν, and (b) the false positive rate induced by an attack is bounded by α. Both constraints can be motivated by an operational practice of anomaly detection systems. Overall, we significantly extend the analysis of Nelson and Joseph (2006) by considering a more realistic learning scenario, explicitly treating potential constraints on the attacker's part and providing tighter bounds.

Our analysis methodology follows the following framework, which can be used for any kind of *quantitative security analysis* of learning algorithms (Laskov and Kloft, 2009):

1. *Axiomatic formalization of the learning and the attack processes*. Such a formalization includes definitions of data sources and objective (risk) functions used by each party, as well as the attacker's goal. It specifies the knowledge available to an attacker, that is, whether he knows an algorithm, its parameters and internal state, and which data he can potentially manipulate.

2. *Specification of an attacker's constraints*. Potential constraints on the attacker's part may include: percentage of traffic under his control, amount of additional data that must be injected, an upper bound on the norm of a manipulated part, a maximal allowable false-positive rate (in case an attack must be stealthy), etc. Such constraints must be incorporated into the axiomatic formalization.

3. *Investigation of the optimal attack policy*. Such a policy may be long-term; that is, over multiple attack iterations, as well as short-term, for a single iteration. The investigation can be carried out either as a formal proof or numerically, by casting the search for an attack policy as an optimization problem.

4. *Bounding the attacker's gain under the optimal policy*. The ultimate goal of our analysis is to quantify the attacker's gain or effort under his optimal policy. Such an analysis may take different forms, for example calculation of the probability for an attack to succeed, estimation of the required number of attack iterations, calculation of the geometric impact of an attack (a shift towards an insecure state), etc.

Organization of this paper reflects the main steps of the proposed methodology. In the preliminary Section 2, the models of the learning and attack processes are introduced. The analytical part is arranged in two sections. In Sections 3 and 4, we address the steps (1), (3) and (4) under the assumption that the attacker has full control of the network traffic, first assuming that a learning algorithm can memorize all previously seen examples, followed by the finite memory case. Section 5 introduces the assumption that the attacker's control is limited to a fixed fraction of network traffic, as required in step (2). Another constraint (bounded false positive rate) is considered in Section 6. This section also removes the somewhat unrealistic assumption of Section 5 that all innocuous points are accepted by the algorithm. Analytic results are experimentally verified in Section 8 on real HTTP data and attacks used in intrusion detection systems. Some proofs and auxiliary technical material are presented in the Appendix. The notation used in the paper is summarized in Table 1.

r	centroid's radius
i	attack iteration index, $i \in \mathbb{N}_0$
$\mathbf{c}_i$	center of centroid in i-th attack iteration
$\mathbf{A}$	attack point
$\mathbf{a}$	attack direction vector
D_i	i-th relative displacement of a centroid in radii into direction of $\mathbf{a}$
n	number of patterns used for initial training of the centroid
f	attack strategy function
ν	fraction of adversarial training points
B_i	Bernoulli random variable
$\mathbf{x}_i$	training data
α	false alarm rate
I_S	indicator function of a set S

Table 1: Notation summary.

1.2 Poisoning and Related Attacks Against Learning Algorithms

For two-class learning problems, attacks against learning algorithms can be generally classified according to the following two criteria (the terminology in the taxonomy of Barreno et al. (2006) is given in brackets):

- whether an attack is staged during the training (causative) or the deployment of an algorithm (exploratory), or
- whether an attack attempts to increase the false negative or the false positive rate at the deployment stage (integrity/availability).

The poisoning attack addressed in our work can be classified as a causative integrity attack. This scenario is quite natural, for example, in web application scenarios in which the data on a server can be assumed to be secure but the subsequent injection of adversarial data cannot be easily prevented. Other common attack types are the mimicry attack—alteration of malicious data to resemble innocuous data (an exploratory integrity attack) and the "red herring" attack—sending junk data that causes false alarms (an exploratory availability attack). Attacks of the latter two kinds are beyond the scope of our investigation.

As a final remark, we must consider the extent to which the attacker is familiar with the learning algorithm and trained model. One of the key principles of computer security, known as *Kerckhoff's principle*, is that the robustness of any security instrument must not depend on keeping its operational functionality secret. Similar to modern cryptographic methods, we must assume that the attacker knows which machine learning algorithm is deployed and how it operates (he can even use machine learning to reverse engineer deployed classifiers, as shown by Lowd and Meek, 2005b). However, it may be more difficult for an attacker to obtain the training data or the particular learned model. In the case of anomaly detection, it is relatively easy for an attacker to retrieve the learned model: it suffices to sniff on the same application that is protected by an algorithm to get approxi-

mately the same innocuous data the algorithm is trained on. Hence, we will assume that the attacker has precise knowledge of the trained model throughout the attack.

2. Learning and Attack Models

Before proceeding with the analysis, we first present the precise models of the learning and the attack processes.

2.1 Centroid Anomaly Detection

Given a data set $X_0 = \{\mathbf{x}_0^{(1)}, \ldots, \mathbf{x}_0^{(n)}\} \subset \mathbb{R}^d$, the goal of anomaly detection (also often referred to as "novelty detection") is to determine whether an example $\mathbf{x}$ is unlikely to have been generated by the same distribution as the set X_0. A natural way to perform anomaly detection is to estimate the probability density function of the distribution from which the set X_0 was drawn and mark $\mathbf{x}$ as anomalous if it comes from a region with low density. In general, however, density estimation is a difficult problem, especially in high dimensions. A large amount of data is usually needed to reliably estimate the density in all regions of the space. For anomaly detection, knowing the density in the entire space is superfluous, as we are only interested in deciding whether a specific point lies within a "sparsely populated" area. Hence several direct methods have been proposed for anomaly detection, for example, one-class SVM (Schölkopf et al., 2001), support vector data description (SVDD) (Tax and Duin, 1999a,b), and density level set estimation (Polonik, 1995; Tsybakov, 1997; Steinwart et al., 2005). A comprehensive survey of anomaly detection techniques can be found in Markou and Singh (2003a,b).

In the centroid anomaly detection, an Euclidean distance from the empirical mean of the data is used as a measure of abnormality:

$$f(\mathbf{x}) = \left|\left|\mathbf{x} - \frac{1}{n}\sum_{j=1}^{n} \mathbf{x}_0^{(j)}\right|\right|.$$

If a hard decision is desired instead of a soft abnormality score, the data point is considered anomalous if its score exceeds a fixed threshold r.

Despite its straightforwardness, a centroid model can represent complex density level sets using a kernel mapping (Müller et al., 2001; Schölkopf and Smola, 2002) (see Figure 1). Centroid anomaly detection can be seen as a special case of the SVDD with outlier fraction $\eta = 1$ and of the Parzen window density estimator (Parzen, 1962) with Gaussian kernel function $k(\mathbf{x},\mathbf{y}) = \frac{1}{(\sqrt{2\pi}\sigma)^d}\exp(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{y}\|^2)$.

Centroid anomaly detection has been successfully used in a variety of anomaly detection applications such as intrusion detection (Hofmeyr et al., 1998; Yeung and Chow, 2002; Laskov et al., 2004a; Wang and Stolfo, 2004; Rieck and Laskov, 2006; Wang et al., 2006; Rieck and Laskov, 2007), wireless sensor networks (Rajasegarar et al., 2007) and jet engine vibration data analysis (Nairac et al., 1999). It has been shown (see, for example, Section 4.1 in Shawe-Taylor and Cristianini, 2004; Vert and Vert, 2006) that even in high-dimensional spaces induced by nonlinear feature maps, the empirical estimator of the center of mass of the data is stable and the radius of the sphere anchored at the center of mass is related to the level set of the corresponding probability density.

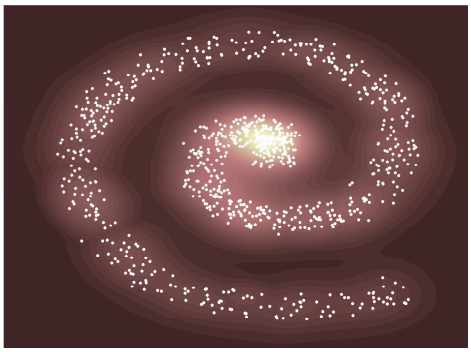


Figure 1: Illustration of the density level estimation using a centroid model with a non-linear kernel.

2.2 Online Anomaly Detection

The majority of anomaly detection applications have to deal with non-stationary data. This is especially typical for computer security, as the processes being monitored usually change over time. For example, network traffic characteristics are strongly influenced by the time of the day; system call sequences depend on the applications running on a computer. The model of normality constructed by anomaly detection algorithms hence needs to be regularly updated, in the extreme case—after the arrival of each data point. Obviously, retraining the model from scratch every time is computationally inefficient; however, the incorporation of new data points and the removal of irrelevant ones can be done with acceptable effort (e.g., Laskov et al., 2006).

For centroid anomaly detection, we assume that the initial state of the learner comprises a center $\mathbf{c}_0$ and a given radius r. We further assume that this state has been obtained from purely innocuous data. Whenever a new data point $\mathbf{x}_i$ arrives at iteration $i \in \mathbb{N}$, the learner's center of mass $\mathbf{c}_i$ is updated if and only if the new data point is considered non-anomalous; otherwise, it is rejected and not used for re-training. The radius r stays fixed over time.[2] Recalculation of the center of mass is straightforward and requires $O(1)$ work. If all examples are "memorized", that is, $X_i = X_{i-1} \cup \{x_i\}$,[3] the update is computed as

$$\mathbf{c}_{i+1} = \left(1 - \frac{1}{n+i}\right)\mathbf{c}_i + \frac{1}{n+i}\mathbf{x}_i\,. \tag{1}$$

For a finite horizon, that is, $\forall i : |X_i| = n$, in each iteration i, some previous example $\mathbf{x}_{\text{old}} \in X_i$ is replaced by the newly arriving $\mathbf{x}_i$, and the update is thus performed as

$$\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n}(\mathbf{x}_i - \mathbf{x}_{\text{old}})\,. \tag{2}$$

The update formula can be generalized to $\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{\kappa}{i}(\mathbf{x}_i - \mathbf{x}_{\text{old}})$, with fixed $\kappa \geq 0$. This changes the bounds in the upcoming analysis only by a constant factor in this case, which is negligible.

2. From the operational standpoint, this assumption is reasonable as the radius serves as a threshold and is usually set and manually tuned after inspection of (false) alarms. Treating the radius as a model parameter would necessitate complex rules for its update and would open the system to poisoning attacks against the radius.
3. Note that the data need not be physically stored, as the contributions from individual data points are accumulated in a current location of the center.

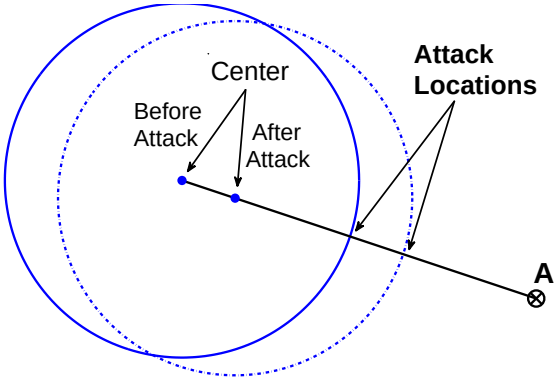


Figure 2: Illustration of a poisoning attack. By iteratively inserting malicious training points an attacker can gradually "drag" the centroid into a direction of an attack. The radius is assumed to stay fixed over time. Figure taken from Nelson and Joseph (2006).

Various strategies can be used to determine the "least relevant" point $\mathbf{x}_{\text{old}}$ to be removed from the working set:

(a) oldest-out: the point with the oldest timestamp is removed.

(b) random-out: a randomly chosen point is removed.

(c) nearest-out: the nearest-neighbor of the new point $\mathbf{x}$ is removed.

(d) average-out: the center of mass is removed. The new center of mass is recalculated as $\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n}(\mathbf{x}_i - \mathbf{c}_i)$, which is equivalent to Equation (1) with constant n.

The strategies (a)–(c) require the storage of all points in the working set, whereas the strategy (d) can be implemented by holding only the center of mass in memory.

2.3 Poisoning Attack

The goal of a poisoning attack is to force an algorithm, at some learning iteration i, to accept the attack point $\mathbf{A}$ that lies outside of the normality ball, that is, $||\mathbf{A} - \mathbf{c}_i|| > r$. It is assumed that the attacker knows the anomaly detection algorithm and the training data. Furthermore, the attacker cannot modify any existing data except for adding new points. Although the attacker can send any data point, it obviously only makes sense for him to send points lying within the current sphere as they are otherwise discarded by the learning algorithm. As illustrated in Figure 2, the poisoning attack amounts to injecting specially crafted points that are accepted as innocuous but shift the center of mass in the direction of the attack point until the latter appears innocuous as well.

What points should be used by an attacker in order to subvert online anomaly detection? Intuitively, one can expect that the optimal one-step displacement of the center of mass is achieved by placing attack point $\mathbf{x}_i$ along the line connecting $\mathbf{c}$ and $\mathbf{A}$ such that $||\mathbf{x}_i - \mathbf{c}|| = r$. A formal proof of the optimality of such a strategy and the estimation of its efficiency constitutes the first step of security analysis of online anomaly detection and is provided in the following sections for various scenarios. In our analysis, we will use the following measure of attack effectiveness.

Definition 1 (Relative displacement) *Let* $\mathbf{A}$ *be an attack point and* $\mathbf{a} = \frac{\mathbf{A}-\mathbf{c}_0}{||\mathbf{A}-\mathbf{c}_0||}$ *be the attack direction unit vector. The* i-th relative displacement D_i *of an online centroid learner is defined as*

$$D_i = \frac{(\mathbf{c}_i - \mathbf{c}_0)\cdot\mathbf{a}}{r}\,.$$

The relative displacement measures the length of the projection of accrued change to $\mathbf{c}_i$ onto the attack direction $\mathbf{a}$ in terms of the radius of the normality ball. Note that the displacement is a relative quantity, that is, we may without loss of generality translate the coordinate system so that the center of mass lies at the origin (i.e., $\mathbf{c}_0 = \mathbf{0}$) and subsequently isotropically normalize the space so that the centroid has unit radius $r = 1$. After this transformation, the formula for the displacement can be simplified to

$$D_i = \mathbf{c}_i \cdot \mathbf{a}\,.$$

Definition 2 *An attack strategy that maximizes the displacement* D_i *in each iteration* i *is referred to as* greedy-optimal.

3. Attack Effectiveness for Infinite Horizon Centroid Learner

The effectiveness of a poisoning attack for the infinite horizon learner has been analyzed in Nelson and Joseph (2006). We provide an alternative proof that follows the main steps of the framework proposed in Section 1.1.

Theorem 3 *The* i*-th relative displacement* D_i *of the online centroid learner with an infinite horizon under a poisoning attack is bounded by*

$$D_i \leq \ln\left(1 + \frac{i}{n}\right), \tag{3}$$

where i *is the number of attack points and* n *is the number of initial training points.*

Proof We first determine the greedy-optimal attack strategy and then bound the attack progress.

(a) Let $\mathbf{A}$ be an attack point and denote by $\mathbf{a}$ the corresponding attack direction vector. Let $\{\mathbf{x}_i | i \in \mathbb{N}\}$ be adversarial training points. The center of mass at the ith iteration is given in the following recursion:

$$\mathbf{c}_{i+1} = \left(1 - \frac{1}{n+i}\right)\mathbf{c}_i + \frac{1}{n+i}\mathbf{x}_{i+1}\,, \tag{4}$$

with initial value $\mathbf{c}_0 = 0$. By the construction of the poisoning attack, $||\mathbf{x}_i - \mathbf{c}_i|| \leq r$, which is equivalent to $\mathbf{x}_i = \mathbf{c}_i + \mathbf{b}_i$ with $||\mathbf{b}_i|| \leq r$. Equation (4) can thus be transformed into

$$\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n+i}\mathbf{b}_i\,.$$

Taking the scalar product with $\mathbf{a}$ and using the definition of a relative displacement, we obtain:

$$D_{i+1} = D_i + \frac{1}{n+i}\cdot\frac{\mathbf{b}_i\cdot\mathbf{a}}{r}\,, \tag{5}$$

with $D_0 = 0$. The right-hand side of the Equation (5) is clearly maximized under $||\mathbf{b}_i|| \leq 1$ by setting $\mathbf{b}_i = r\mathbf{a}$. Thus the greedy-optimal attack is given by

$$\mathbf{x}_i = \mathbf{c}_i + r\mathbf{a}\,. \tag{6}$$

(b) Plugging the optimal strategy $\mathbf{b}_i = r\mathbf{a}$ into Equation (5), we obtain:

$$D_{i+1} = D_i + \frac{1}{n+i}\,.$$

This recursion can be explicitly solved, taking into account that $D_0 = 0$, resulting in:

$$D_i = \sum_{k=1}^{i} \frac{1}{n+k} = \sum_{k=1}^{n+i} \frac{1}{k} - \sum_{k=1}^{n} \frac{1}{k}\,.$$

Inserting the upper bound on the harmonic series, $\sum_{k=1}^{m} \frac{1}{k} = \ln(m) + \varepsilon_m$, with $\varepsilon_m \geq 0$, into the above formula and noting that ε_m is monotonically decreasing, we obtain:

$$D_i \leq \ln(n+i) - \ln(n) = \ln\left(\frac{n+i}{n}\right) = \ln\left(1 + \frac{i}{n}\right)\,,$$

which completes the proof. ■

Since the bound in Equation (3) is monotonically increasing, we can invert it to obtain a bound on the effort needed by an attacker to achieve his goal:

$$i \geq n \cdot (\exp(D_i) - 1)\,.$$

It can be seen that the effort needed to poison an online centroid learner is *exponential* in terms of the relative displacement of the center of mass.[4] In other words, an attacker's effort grows prohibitively fast with respect to the separability of the attack from innocuous data. For a kernelized centroid learner, the greedy-optimal attack may not be valid, as there may not exist a point in the input space corresponding to the optimal attack image in the feature space. However, an attacker can construct points in the input space that are close enough to the greedy-optimal point for the attack to succeed, with a moderate constant cost factor; cf., Section 8.5.

4. Poisoning Attack against Finite Horizon Centroid Learner

The optimistic result presented in Section 3 is unfortunately not quite useful. In practice, memorization of all training points essentially defeats the main purpose of an online algorithm, that is, its ability to adjust to non-stationarity. Hence it is important to understand how the removal of data points from the working set affects the security of online anomaly detection. To this end, the specific removal strategies presented in Section 2.2 must be considered. The analysis can be carried out theoretically for the average-out and random-out update rules; for the nearest-out rule, an optimal attack can be stated as an optimization problem and the attack effectiveness can be analyzed empirically.

4. Even constraining a maximum number of online update steps does not remove this bound's exponential growth (Nelson and Joseph, 2006).

4.1 Poisoning Attack for Average-out, Random-out and Oldest-out Rules

We begin our analysis with the average-out learner which follows exactly the same update rule as the infinite-horizon online centroid learner with the exception that the window size n remains fixed instead of growing indefinitely (cf. Section 2.2). Despite the similarity to the infinite-horizon case, the result presented in the following theorem is surprisingly pessimistic.

Theorem 4 *The i-th relative displacement D_i of the online centroid learner with the average-out update rule under a worst-case optimal poisoning attack is*

$$D_i = \frac{i}{n},$$

where i is the number of attack points and n is the training window size.

Proof The proof is similar to the proof of Theorem 3. By explicitly writing out the recurrence between subsequent displacements, we conclude that the greedy-optimal attack is also attained by placing an attack point along the line connecting $\mathbf{c}_i$ and $\mathbf{A}$ at the edge of the sphere (cf. Equation (6)):

$$\mathbf{x}_i = \mathbf{c}_i + r\mathbf{a}\,.$$

It follows that the relative displacement under the greedy-optimal attack is

$$D_{i+1} = D_i + \frac{1}{n}\,.$$

Since this recurrence is independent of the running index i, the displacement is simply accumulated over each iteration, which yields the bound of the theorem. ■

One can see that, unlike the logarithmic bound in Theorem 3, the average-out learner is characterized by a linear bound on the displacement. As a result, an attacker only needs a linear number of injected points—instead of an exponential one—in order to subvert an average-out learner. This cannot be considered secure.

A similar result, in terms of the *expectation* of the relative displacement, can be obtained for the random-out removal strategy. The proof is based on the observation that in expectation, the average-out rule is equivalent to the random-out rule. The oldest-out rule can also be handled similarly to the average-out rule by observing that in both cases some fixed point known in advance is removed from a working set, which allows an attacker to easily find an optimal attack point.

4.2 Poisoning Attack for Nearest-out Rule

One might expect that the nearest-out strategy poses a stronger challenge to an attacker as it tries to retain working set diversity by eliminating the most similar data to the new point. However, even this strategy can be broken with a feasible amount of work if the attacker follows a greedy-optimal strategy. The latter is the subject of our investigation in this section.

4.2.1 GREEDY-OPTIMAL ATTACK

The *greedy-optimal* attack should provide a maximal gain for an attacker in a single iteration. For the infinite-horizon learner (and hence also for the average-out learner, as it uses the same recurrence in a proof), it is possible to show that the greedy-optimal attack yields the maximum gain for the entire sequence of attack iterations; that is, it is (globally) optimal. For the nearest-out learner, it is hard to analyze the full sequence of attack iterations, hence we limit our analysis to a single-iteration gain. Empirically, even a greedy-optimal attack turns out to be effective.

To construct a greedy-optimal attack, we partition the sphere spanned by the centroid into Voronoi cells V_j centered at the training data points $\mathbf{x}_j$, $j = 1,\dots,n$. Each Voronoi cell comprises points for which $\mathbf{x}_j$ is the nearest neighbor. Whenever a new training point "falls into" the sphere, the center of the corresponding Voronoi cell is removed according to the nearest-out rule.

The optimal attack strategy is now straightforward. First, we determine the optimal attack location within each cell. This can be done by solving the following optimization problem for a fixed $\mathbf{x}_j$:

Optimization Problem 5 (greedy-optimal attack)

$$\mathbf{x}_j^* = \operatorname*{argmax}_{\mathbf{x}} \; y_j(\mathbf{x}) := (\mathbf{x}-\mathbf{x}_j)\cdot\mathbf{a} \tag{7}$$

$$s.t. \;\; \|\mathbf{x}-\mathbf{x}_j\| \le \|\mathbf{x}-\mathbf{x}_k\|, \quad \forall k = 1,...,n \tag{8}$$

$$\|\mathbf{x}-\tfrac{1}{n}\textstyle\sum_{k=1}^n \mathbf{x}_k\| \le r\,. \tag{9}$$

The objective of the optimization problem (7) reflects the goal of maximizing the projection of $\mathbf{x}-\mathbf{x}_j$ onto the attack direction $\mathbf{a}$. The constraint (8) stipulates that the point $\mathbf{x}_j$ is the nearest neighbor of $\mathbf{x}$. The constraint (9), when active, enforces that no solution lies outside of the sphere.

The geometry of the greedy-optimal attack is illustrated in Figure 3. An optimal attack point is placed at the "corner" of a Voronoi cell (including possibly a round boundary of the centroid) to cause the largest displacement of the centroid along the attack direction.

Once the candidate attack locations are found for each of the n Voronoi cells, the one that has the highest value of the objective function $y_j(\mathbf{x}_j^*)$ is injected and the respective center $\mathbf{x}_{j^*}$ of the Voronoi cell is expunged from the training set:

$$j^* = \operatorname{argmax}_{j\in 1,\dots,n} y_j(\mathbf{x}_j^*)\,. \tag{10}$$

The optimization problem (7) can be simplified by plugging in the definition of the Euclidean norm. In the resulting optimization problem, all but one of the norm constraints are reduced to simpler linear constraints:

$$\begin{aligned} \mathbf{x}_j^* = \operatorname*{argmax}_{\mathbf{x}} \; & (\mathbf{x}-\mathbf{x}_j)\cdot\mathbf{a} \\ \text{s.t. } & 2(\mathbf{x}_k-\mathbf{x}_j)\cdot\mathbf{x} \le \mathbf{x}_k\cdot\mathbf{x}_k - \mathbf{x}_j\cdot\mathbf{x}_j, \quad \forall k = 1,...,n \\ & \mathbf{x}\cdot\mathbf{x} - \tfrac{2}{n}\textstyle\sum_{k=1}^n \mathbf{x}\cdot\mathbf{x}_k \le r^2 - \frac{1}{n^2}\sum_{k,l=1}^n \mathbf{x}_k\cdot\mathbf{x}_l\,. \end{aligned} \tag{11}$$

Due to the quadratic constraint, the inner optimization task is not as simple as a linear or a quadratic program. However, several standard optimization packages; for example, CPLEX or MOSEK,

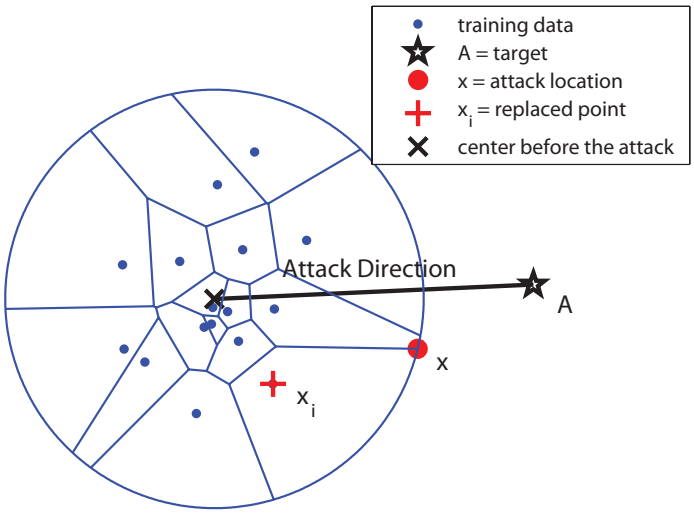


Figure 3: The geometry of a poisoning attack for the nearest-out rule. A greedy-optimal attack is injected at the boundary of the respective Voronoi cell.

can optimize such quadratically constrained linear programs (QCLP) with high efficiency, especially when there is only a single quadratic constraint. Alternatively, one can use specialized algorithms for linear programming with a single quadratic constraint (van de Panne, 1966; Martein and Schaible, 2005) or convert the quadratic constraint to a second-order cone (SOC) constraint and use general-purpose conic optimization methods.

4.2.2 IMPLEMENTATION OF A GREEDY-OPTIMAL ATTACK

For the practical implementation of the attack specified by problem (11), some additional processing steps must be carried out.

A point can become "immune" to a poisoning attack, if starting from some iteration i' its Voronoi cell does not overlap with the hypersphere of radius r centered at $\mathbf{c}_{i'}$. The quadratic constraint (9) is never satisfied in this case, and the inner optimization problem (7) becomes infeasible. These immune points remain in the working set forever and slow down the attack's progress. To avoid this situation, an attacker must keep track of all optimal solutions $\mathbf{x}_j^*$ of the inner optimization problems. If an online update would cause some Voronoi cell V_j to completely slip out of the hypersphere an attacker should ignore the outer loop decision (10) and expunge $\mathbf{x}_j$ instead of $\mathbf{x}_{j^*}$.

A significant speedup can be attained by avoiding the solution of unnecessary QCLP problems. Let $S = \{1,\ldots,j-1\}$ and α_S be the current best solution of the outer loop problem (10) over the set S. Let y_{α_S} be the corresponding objective value of an inner optimization problem (11). Consider the following auxiliary quadratic program (QP):

$$\max_{\mathbf{x}} \ \|\mathbf{x} - \tfrac{1}{n}\textstyle\sum_{k=1}^{n} \mathbf{x}_k\| \tag{12}$$

$$\text{s.t. } 2(\mathbf{x}_k - \mathbf{x}_j)\cdot\mathbf{x} \le \mathbf{x}_k\cdot\mathbf{x}_k - \mathbf{x}_j\cdot\mathbf{x}_j, \quad \forall k = 1,\ldots,n \tag{13}$$

$$(\mathbf{x} - \mathbf{x}_j)\cdot\mathbf{a} \ge y_{\alpha_S}\,. \tag{14}$$

Its feasible set comprises the Voronoi cell of $\mathbf{x}_j$, defined by constraint (13), further reduced by constraint (14) to the points that improve the current value y_{α_s} of the global objective function. If the objective function value provided by the solution of the auxiliary QP (12) exceeds r then the solution of the local QCLP (11) does not improve the global objective function y_{α_s}. Hence the expensive QCLP optimization can be skipped.

4.2.3 ATTACK EFFECTIVENESS

To evaluate the effectiveness of the greedy-optimal attack, we perform a simulation on artificial geometric data. The goal of this simulation is to investigate the behavior of the relative displacement D_i during the progress of the greedy-optimal attack.

An initial working set of size $n = 100$ is sampled from a d-dimensional Gaussian distribution with unit covariance (experiments are repeated for various values of $d \in \{2, ..., 100\}$). The radius r of the online centroid learner is chosen such that the expected false positive rate is bounded by $\alpha = 0.001$. An attack direction $\mathbf{a}$, $\|\mathbf{a}\| = 1$, is chosen randomly, and 500 attack iterations ($5 * n$) are generated using the procedure presented in Sections 4.2.1–4.2.2. The relative displacement of the center in the direction of the attack is measured at each iteration. For statistical significance, the results are averaged over 10 runs.

Figure 4(a) shows the observed progress of the greedy-optimal attack against the nearest-out learner and compares it to the behavior of the theoretical bounds for the infinite-horizon learner (the bound of Nelson and Joseph, 2006) and the average-out learner. The attack effectiveness is measured for all three cases by the relative displacement as a function of the number of iterations. Plots for the nearest-out learner are presented for various dimensions d of the artificial problems tested in simulations. The following observations can be made from the plot provided in Figure 4(a).

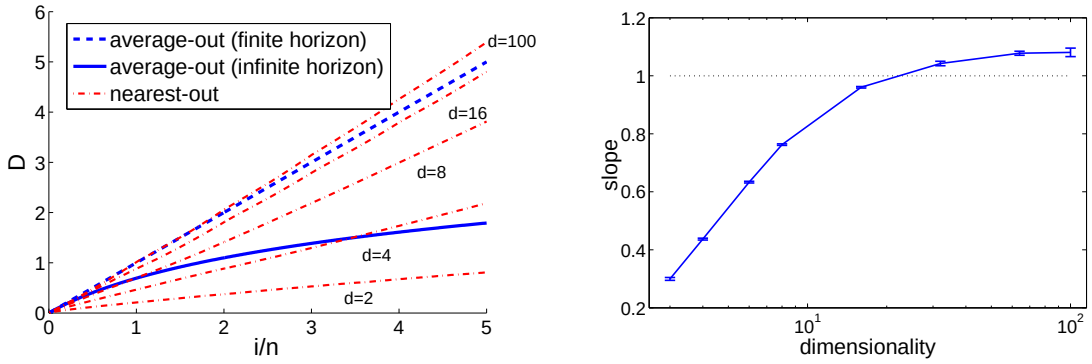
First, the attack progress, that is, the functional dependence of the relative displacement of the greedy-optimal attack against the nearest-out learner with respect to the number of iterations, is *linear*. Hence, contrary to our initial intuition, the removal of nearest neighbors of incoming points does not lead to better security against poisoning attacks.

Second, the slope of the linear attack progress *increases with the dimensionality of the problem*. For low dimensionality, the relative displacement of the nearest-out learner is comparable, in absolute terms, with that of the infinite-horizon learner. For high dimensionality, the nearest-out learner becomes even less secure than the simple average-out learner. By increasing the dimensionality beyond $d > n$ the attack effectiveness cannot be increased. Mathematical reasons for this behavior are investigated in Section A.1.

A further illustration of the behavior of the greedy-optimal attack is given in Figure 4(b), showing the dependence of the average attack slope on the dimensionality. One can see that the attack slope increases logarithmically with the dimensionality and wanes out to a constant factor after the dimensionality exceeds the number of training data points. A theoretical explanation of the observed experimental results is given in Appendix A.1.2.

4.3 Concluding Remarks

To summarize our analysis for the case of the attacker's full control over the training data, we conclude that an optimal poisoning attack successfully subverts a finite-horizon online centroid learner for all outgoing point selection rules. This conclusion contrasts with the analysis of the infinite-horizon learner carried out in Barreno et al. (2006) that yields a logarithmic attack progress.



(a) (b)

Figure 4: Effectiveness of the poisoning attack for the nearest-out rule as a function of input space dimensionality. (a) The displacement of the centroid along the attack direction grows linearly with the number of injected points. Upper bounds on the displacement of the average-out rule are plotted for comparison. (b) The slope of the linear growth increases with the input space dimensionality.

As a compromise, one can in practice choose a large working set size n, which reduces the slope of a linear attack progress.

Among the different outgoing point selection rules, the nearest-out rule presents the most challenges for the implementation of an optimal attack. Some approximations make such an attack feasible while still maintaining a reasonable progress rate. The key factor for the success of a poisoning attack in the nearest-out case lies in high dimensionality of the feature space. The progress of an optimal poisoning attack depends on the size of the Voronoi cells induced by the training data points. The size of the Voronoi cells is related linearly to the volume of the sphere corresponding to the attack's feasible region (see Appendix A.1.2 for a theoretical discussion of this effect). With the increasing dimensionality of the feature space, the volume of the sphere increases exponentially, which leads to a higher attack progress rate.

In the following sections, we analyze two additional factors that affect the progress of a poisoning attack. First, we consider the case when the attacker controls only a fixed fraction ν of the training data. Subsequently we analyze a scenario in which an attacker is not allowed to exceed a certain false positive rate α; for example, by stopping online learning when a high false positive rate is observed. It will be shown that both of these possible constraints significantly reduce the effectiveness of poisoning attacks.

5. Poisoning Attack with Limited Bandwidth Constraint

Until now we assumed that an attacker has unlimited control over the training data; that is, in the worst case, all the data seen by the learner may be adversarial. This assumption is too pessimistic, as typically a deployed learning algorithm would also receive normal data during its operation. We would now analyze centroid anomaly detection under the assumption that only a fraction ν of the training data is adversarial. Our goal is to analyze the impact of this fraction of control on the

difficulty of an attack. The choice of realistic values of ν is application-specific. For simplicity, we restrict ourselves to the average-out learner as we have seen that it only differs by a constant from the nearest-out one and is equivalent in expectation to the random-out one.

5.1 Learning and Attack Models

As discussed in Section 2, we assume that the initial online centroid learner is centered at the position $\mathbf{c}_0$ and has a fixed radius r (without loss of generality $\mathbf{c}_0 = \mathbf{0}$ and $r = 1$; cf. discussion after Definition 1 in Section 2). At each iteration a new training point arrives—which is either inserted by an adversary or drawn independently from the distribution of innocuous points—and a new center of mass $\mathbf{c}_i$ is calculated. The mixing of innocuous and attack points is modeled by a Bernoulli random variable with the parameter ν which denotes the probability that an adversarial point is presented to the learner. Adversarial points $\mathbf{A}_i$ are chosen according to the attack function f depending on the actual state of the learner $\mathbf{c}_i$. The innocuous pool is modeled by a probability distribution from which the innocuous points $\mathbf{x}_i$ are independently drawn. We assume that the expectation of innocuous points $\mathbf{x}_i$ coincides with the initial center of mass: $E(\mathbf{x}_i) = \mathbf{c}_0$.

For simplicity, we make one additional assumption in this chapter: *all innocuous points are accepted by the learner at any time of the attack independent of their actual distance to the center of mass*. In the absence of this assumption, we would need special treatment for the case that a truly innocuous point is disregarded by the learner as the center of mass is getting displaced by the attack. In the next section we drop this assumption so that the learner only accepts points which fall within the actual radius.

The described probabilistic model is formalized by the following axiom.

Axiom 6 *$\{B_i | i \in \mathbb{N}\}$ are independent Bernoulli random variables with parameter $\nu > 0$. $\mathbf{x}_i$ are i.i.d. random variables in a Euclidean space $\mathbb{R}^d$, drawn from a fixed but unknown distribution $P_{\mathbf{x}}$, satisfying $E(\mathbf{x}) = \mathbf{0}$ and $\|\mathbf{x}\| \leq r \overset{w.l.o.g.}{=} 1$. B_i and $\mathbf{x}_j$ are mutually independent for each i, j. $f : \mathbb{R}^d \to \mathbb{R}^d$ is a function $\|f(\mathbf{x}) - \mathbf{x}\| \leq r$ that we call* attack strategy. *$\{\mathbf{c}_i | i \in \mathbb{N}\}$ is a collection of random vectors such that w.l.o.g. $\mathbf{c}_0 = 0$ and*

$$\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n} \left(B_i f(\mathbf{c}_i) + (1 - B_i)\mathbf{x}_i - \mathbf{c}_i \right) . \tag{15}$$

Moreover, we denote $x_i := \mathbf{x}_i \cdot \mathbf{a}$.

For simplicity of notation, in this section we refer to a collection of random vectors $\{\mathbf{c}_i | i \in \mathbb{N}\}$ satisfying Axiom 6 as an *online centroid learner*. Any function f satisfying Axiom 6 is called an *attack strategy*. The attack strategy is a function that maps a vector (the center) to an attack location.

According to the above axiom, the adversary's attack strategy is formalized by an *arbitrary* function f. This raises the question of which attack strategies are optimal in the sense that an attacker reaches his goal of concealing a predefined attack direction vector in a minimal number of iterations. As in the previous sections, an attack's progress is measured by projecting the current center of mass onto the attack direction vector:

$$D_i = \mathbf{c}_i \cdot \mathbf{a} .$$

Attack strategies maximizing the displacement D_i in each iteration i are referred to as *greedy-optimal attack strategies*.

5.2 Greedy-Optimal Attack

The following result characterizes the greedy-optimal attack strategy for the model specified in Axiom 6.

Proposition 7 *Let* $\mathbf{a}$ *be an attack direction vector. Then the* greedy-optimal attack strategy f *against the online centroid learner is given by*

$$f(\mathbf{c}_i) := \mathbf{c}_i + \mathbf{a}\,. \tag{16}$$

Proof Since by Axiom 6 we have $\|f(\mathbf{x}) - \mathbf{x}\| \leq r$, any valid attack strategy can be written as $f(\mathbf{x}) = \mathbf{x} + g(\mathbf{x})$, such that $\|g\| \leq r = 1$. It follows that

$$\begin{aligned} D_{i+1} &= \mathbf{c}_{i+1} \cdot \mathbf{a} \\ &= \left(\mathbf{c}_i + \frac{1}{n}\left(B_i f(\mathbf{c}_i) + (1-B_i)\mathbf{x}_i - \mathbf{c}_i\right)\right) \cdot \mathbf{a} \\ &= D_i + \frac{1}{n}(B_i D_i + B_i g(\mathbf{c}_i) \cdot \mathbf{a} + (1-B_i)\mathbf{x} \cdot \mathbf{a} - D_i)\,. \end{aligned}$$

Since $B_i \geq 0$, the greedy-optimal attack strategy should maximize $g(\mathbf{c}_i) \cdot \mathbf{a}$ subject to $||g(\mathbf{c}_i)|| \leq 1$. The maximum is clearly attained by setting $g(\mathbf{c}_i) = \mathbf{a}$. ■

Note that the displacement measures the projection of the change of the centroid onto the attack direction vector. Hence it is not surprising that the optimal attack strategy is independent of the actual position of the learner.

5.3 Attack Effectiveness

The effectiveness of the greedy-optimal attack in the limited control case is characterized in the following theorem.

Theorem 8 *For the displacement* D_i *of the centroid learner under an optimal poisoning attack,*

$$\begin{aligned} &\text{(a)} \qquad E(D_i) \;&&=\; (1-a_i)\frac{\nu}{1-\nu} \\ &\text{(b)} \qquad \mathrm{Var}(D_i) \;&&\leq\; \gamma_i \left(\frac{\nu}{1-\nu}\right)^2 + \delta_n\,, \end{aligned}$$

where $a_i := \left(1 - \frac{1-\nu}{n}\right)^i$, $b_i = \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)\right)^i$, $\gamma_i = a_i - b_i$, *and* $\delta_n := \frac{\nu^2 + (1-b_i)}{(2n-1)(1-\nu)^2}$.

Proof (a) Inserting the greedy-optimal attack strategy of Equation (16) into Equation (15) of Axiom 6, we have:

$$\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n}\left(B_i\left(\mathbf{c}_i + \mathbf{a}\right) + (1-B_i)\mathbf{x}_i - \mathbf{c}_i\right)\,,$$

which can be rewritten as:

$$\mathbf{c}_{i+1} = \left(1 - \frac{1-B_i}{n}\right)\mathbf{c}_i + \frac{B_i}{n}\mathbf{a} + \frac{(1-B_i)}{n}\mathbf{x}_i\,. \tag{17}$$

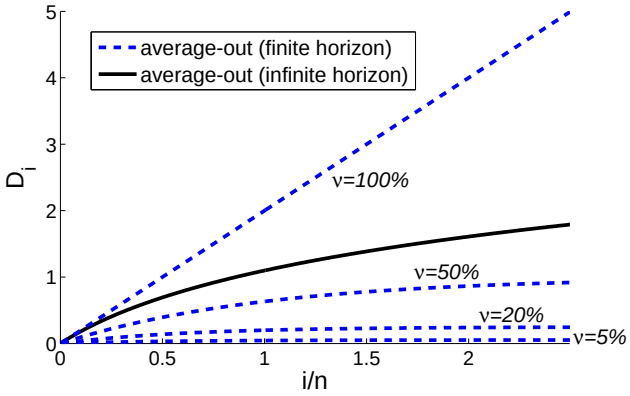


Figure 5: Theoretical behavior of the displacement of a centroid under a poisoning attack for a bounded fraction of traffic under attacker's control. The infinite horizon bound of Nelson et al. is shown for comparison (solid line).

Taking the expectation on the latter equation and noting that by Axiom 6, $E(\mathbf{x}_i) = 0$ and $E(B_i) = \nu$, we have

$$E(\mathbf{c}_{i+1}) = \left(1 - \frac{1-\nu}{n}\right) E(\mathbf{c}_i) + \frac{\nu}{n}\mathbf{a},$$

which by the definition of the displacement translates to

$$E(D_{i+1}) = \left(1 - \frac{1-\nu}{n}\right) E(D_i) + \frac{\nu}{n}.$$

The statement (a) follows from the latter recursive equation by Proposition 17 (formula of the geometric series). For the more demanding proof of (b), see Appendix A.2. ■

The following corollary shows the asymptotic behavior of the above theorem.

Corollary 9 *For the displacement D_i of the centroid learner under an optimal poisoning attack,*

$$\begin{array}{lrcll} \text{(a)} & E(D_i) & \leq & \dfrac{\nu}{1-\nu} & \textit{for all } i \\ \text{(b)} & \mathrm{Var}(D_i) & \to & 0 & \textit{for } i, n \to \infty. \end{array}$$

Proof The corollary follows from the fact that $\gamma_i, \delta_n \to 0$ for $i, n \to \infty$. ■

The behavior of the above bounds as a function of the number of attack iterations is illustrated in Figure 5. One can see that the attack's progress depends on the fraction of the training data controlled by the attacker. For any $\nu < 1$, the attack progress is *bounded by a constant*. Hence the attack's success critically depends on the value of this constant: if $||\mathbf{A} - \mathbf{c}_0|| < \nu/(1-\nu)$, the attack fails *even with infinite effort*. This result provides a much stronger security guarantee than the exponential bound for the infinite horizon case.

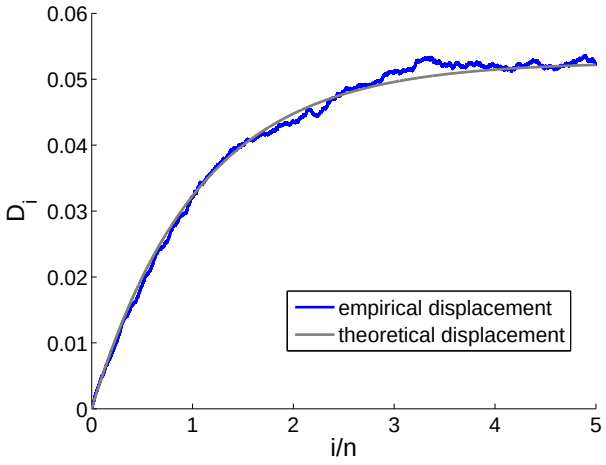


Figure 6: Comparison of the empirical displacement of the centroid under a poisoning attack with attacker's limited control ($\nu = 0.05$) with the theoretical bound for the same setup. The empirical results are averaged over 10 runs.

To empirically investigate the tightness of the derived bound we compute a Monte Carlo simulation of the scenario defined in Axiom 6 with the parameters $\nu = 0.05$, $n = 100000$, $\mathcal{H} = \mathbb{R}^2$, and $P_{\mathbf{x}}$ being a uniform distribution over the unit circle. Figure 6 shows a typical displacement curve over the first $500,000$ attack iterations. One can clearly see that the theoretical bound is closely followed by the empirical simulation.

6. Poisoning Attack under False Positive Constraints

In the last section we have assumed, that innocuous training points $\mathbf{x}_i$ are always accepted by the online centroid learner. It may, however, happen that some innocuous points fall outside of the hypersphere boundary while an attacker displaces the hypersphere. We have seen that the attacker's impact highly depends on the fraction of points he controls. If an attacker succeeds in pushing the hypersphere far enough for innocuous points to start dropping out, the speed of the hypersphere displacement increases. Hence additional protection mechanisms are needed to prevent the success of an attack.

6.1 Learning and Attack Models

Motivated by the above considerations we modify the probabilistic model of the last section as follows. Again we consider the online centroid learner initially anchored at a position $\mathbf{c}_0$ having a radius r. For the sake of simplicity and without loss of generality we can again assume $\mathbf{c}_0 = 0$ and $r = 1$. Innocuous and adversarial points are mixed into the training data according to a fixed fraction controlled by a binary random variable B_i. In contrast to Section 5, innocuous points $\mathbf{x}_i$ are accepted if and only if they fall within the radius r from the hypersphere's center $\mathbf{c}_i$. In addition, to avoid the learner being quickly displaced, we require that the false alarm rate, that is, the number of

innocuous points rejected by the learner, is bounded by α. If the latter is exceeded, we assume the adversary's attack to have failed and a safe state of the learner to be loaded.

We formalize this probabilistic model as follows:

Axiom 10 *$\{B_i | i \in \mathbb{N}\}$ are independent Bernoulli random variables with parameter $\nu > 0$. $\mathbf{x}_i$ are i.i.d. random variables in a reproducing kernel Hilbert space $\mathcal{H}$ drawn from a fixed but unknown distribution $P_{\mathbf{x}}$, satisfying $E(\mathbf{x}) = \mathbf{0}$, $\|\mathbf{x}\| \leq r = 1$, and $P_{\mathbf{x}\cdot\mathbf{a}} = P_{-\mathbf{x}\cdot\mathbf{a}}$ (symmetry w.r.t. the attack direction). B_i and $\mathbf{x}_j$ are mutually independent for each i, j. $f : \mathbb{R}^d \to \mathbb{R}^d$ is an attack strategy satisfying $\|f(\mathbf{x}) - \mathbf{x}\| \leq r$. $\{\mathbf{c}_i | i \in \mathbb{N}\}$ is a collection of random vectors such that $\mathbf{c}_0 = \mathbf{0}$ and*

$$\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n}\left(B_i\left(f(\mathbf{c}_i) - \mathbf{c}_i\right) + (1 - B_i) I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}}\left(\mathbf{x}_i - \mathbf{c}_i\right)\right) ,$$

if $E_{\mathbf{x}}\left(I_{\{\|\mathbf{x} - \mathbf{c}_i\| \leq r\}}\right) \leq 1 - \alpha$ and by $\mathbf{c}_{i+1} = \mathbf{0}$ otherwise. Moreover, we denote $x_i := \mathbf{x}_i \cdot \mathbf{a}$.

For simplicity of notation, we refer to a collection of random vectors $\{\mathbf{c}_i | i \in \mathbb{N}\}$ satisfying Axiom 10 as an *online centroid learner with maximal false positive rate* α in this section. Any function f satisfying Axiom 10 is called an *attack strategy*. Optimal attack strategies are characterized in terms of the displacement as in the previous sections. Note that $E_{\mathbf{x}}(\cdot)$ denotes the conditional expectation given all remaining random quantities except for $\mathbf{x}$.

The intuition behind the symmetry assumption in Axiom 10 is that it ensures that resetting the centroid's center to zero (initiated by the false positive protection) does not lead to a positive shift of the centroid toward the attack direction.

6.2 Greedy-Optimal Attack and Attack Effectiveness

The following result characterizes the greedy-optimal attack strategy for the model specified in Axiom 10. We restrict our analysis to greedy-optimal strategies, that is, the ones that maximize the displacement in each successive iteration.

Proposition 11 *Let $\mathbf{a}$ be an attack direction vector and consider the centroid learner with maximal false positive rate α as defined in Axiom 10. Then the* greedy-optimal *attack strategy f is given by*

$$f(\mathbf{c}_i) := \mathbf{c}_i + \mathbf{a} .$$

Proof Since by Axiom 10 we have $\|f(\mathbf{x}) - \mathbf{x}\| \leq r$, any valid attack strategy can be written as $f(\mathbf{x}) = \mathbf{x} + g(\mathbf{x})$, such that $\|g\| \leq r = 1$. It follows that either $D_{i+1} = 0$, in which case the optimal f is arbitrary, or we have

$$\begin{aligned} D_{i+1} &= \mathbf{c}_{i+1} \cdot \mathbf{a} \\ &= \left(\mathbf{c}_i + \frac{1}{n}\left(B_i f(\mathbf{c}_i) + (1 - B_i)\mathbf{x}_i - \mathbf{c}_i\right)\right) \cdot \mathbf{a} \\ &= D_i + \frac{1}{n}\left(B_i\left(D_i + g(\mathbf{c}_i)\right) + (1 - B_i)\mathbf{x}_i - D_i\right) . \end{aligned}$$

Since $B_i \geq 0$, the greedy-optimal attack strategy should maximize $g(\mathbf{c}_i) \cdot \mathbf{a}$ subject to $||g(\mathbf{c}_i)|| \leq 1$. The maximum is clearly attained by setting $g(\mathbf{c}_i) = \mathbf{a}$. ■

The estimate of effectiveness of the greedy-optimal attack in the limited control case is given in the following theorem.

Theorem 12 *For the displacement D_i of a centroid learner with maximal false positive rate α under a poisoning attack,*

$$\begin{aligned} &\text{(a)} \qquad E(D_i) \quad \leq (1-a_i)\frac{\nu+\alpha(1-\nu)}{(1-\nu)(1-\alpha)} \\ &\text{(b)} \qquad \mathrm{Var}(D_i) \quad \leq \gamma_i \frac{\nu^2}{(1-\alpha)^2(1-\nu)^2} + \rho(\alpha) + \delta_n \,, \end{aligned}$$

where $a_i := \left(1-\frac{(1-\nu)(1-\alpha)}{n}\right)^i$, $b_i = \left(1-\frac{1-\nu}{n}(2-\frac{1}{n})(1-\alpha)\right)^i$, $\gamma_i = (a_i - b_i)$, $\rho(\alpha) = \alpha\frac{(1-a_i)(1-b_i)(2\nu(1-\alpha)+\alpha)}{(1-\frac{1}{2n})(1-\nu)^2(1-\alpha)^2}$, *and* $\delta_n = \frac{(1-b_i)(\nu+(1-\nu)E(x_i^2))}{(2n-1)(1-\nu)(1-\alpha)}$, $x_i := \mathbf{x}_i \cdot \mathbf{a}$.

The proof is technically demanding and is given in Appendix A.3. Despite the more general proof reasoning, we recover the tightness of the bounds of the previous section for the special case of $\alpha = 0$, as shown by the following corollary.

Corollary 13 *Suppose a maximal false positive rate of $\alpha = 0$. Then, the bounds on the expected displacement D_i, as given by Theorem 8 and Theorem 12, coincide. Furthermore, the variance bound of Theorem 12 upper bounds the one of Theorem 8.*

Proof We start by setting $\alpha = 0$ in Theorem 12(a). Clearly the latter bound coincides with its counterpart in Theorem 8. For the proof of the second part of the corollary, we observe that $\rho(0) = 0$ and that the quantities a_i, b_i, and γ_i coincide with their counterparts in Theorem 8. Moreover, removing the distribution dependence by upper bounding $E(x_i) \leq 1$ reveals that δ_i is upper bounded by its counterpart of Theorem 8. Hence, the whole expression on the right hand side of Theorem 12(b) is upper bounded by its counterpart in Theorem 8(b). ■

The following corollary shows the asymptotic behavior of the above theorem. It follows from $\gamma_i, \delta_n, \rho(\alpha) \to 0$ for $i, n \to \infty$, and $\alpha \to 0$, respectively.

Corollary 14 *For the displacement D_i of the centroid learner with maximal false positive rate α under an optimal poisoning attack,*

$$\begin{aligned} &\text{(a)} \qquad E(D_i) \quad \leq \quad \frac{\nu+\alpha(1-\nu)}{(1-\nu)(1-\alpha)} \qquad \textit{for all } i \\ &\text{(b)} \qquad \mathrm{Var}(D_i) \quad \to \; 0 \qquad\qquad\qquad\qquad \textit{for } i, n \to \infty, \alpha \to 0 \,. \end{aligned}$$

From the previous theorem, we can see that for small false positive rates $\alpha \approx 0$, which are common in many applications, for example, intrusion detection (see Section 8 for an extensive empirical analysis), the bound approximately equals the one of the previous section; that is, we have $E(D_i) \leq \frac{\nu}{1-\nu} + \delta$ where $\delta > 0$ is a small constant with $\delta \to 0$. Inverting the bound we obtain the useful formula

$$\nu \geq \frac{E(D_i)}{1+E(D_i)} \,, \tag{18}$$

which gives a lower bound on the minimal ν an adversary has to employ for an attack to succeed.

The illustration of the bound in Theorem 12 is given in Figure 7 for different levels of the false positive protection $\alpha \in [0, 0.025]$. We are especially interested in low false positive rates. One can see that tightness of the bounds of the previous section is almost entirely preserved. In the extreme case $\alpha = 0$, the bounds coincide, as it was shown in Corollary 13.

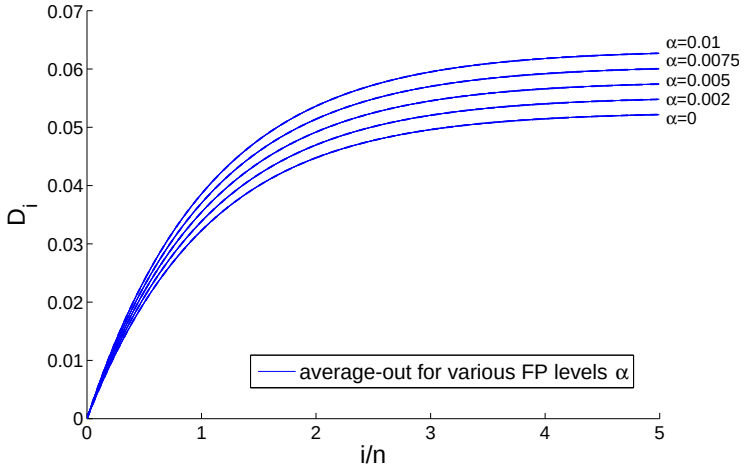


Figure 7: Theoretical behavior of the displacement of a centroid under a poisoning attack for different levels of false positive protection α. The predicted displacement curve for $\alpha = 0$ coincides with the one shown in Figure 6.

7. Generalization to Kernels

For simplicity, we have assumed in the previous sections that the data $\mathbf{x}_i \in \mathbb{R}^d$ lies in an Euclidean space. This assumption does not add any limitations, as all our results can be generalized to the so-called *kernel functions* (Schölkopf and Smola, 2002). This is remarkable, as nonlinear kernels allow one to obtain complex decision functions from the simple centroid model (cf. Figure 1).

Definition 15 (Def. 2.8 in Shawe-Taylor and Cristianini, 2004) *A function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is called a* kernel *if and only if there exists a Hilbert space $(\mathcal{H}, \langle\cdot\rangle)$ and a map $\phi : \mathbb{R}^d \to \mathcal{H}$ such that for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ it holds $k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$. Given a sample $\mathbf{x}_1, \ldots, \mathbf{x}_n$, the matrix K with ij-th entry $k(\mathbf{x}_i, \mathbf{x}_j)$ is called* kernel matrix.

Examination of the theoretical results and proofs of the proceeding sections reveals that they, at no point, require special properties of Euclidean spaces. Instead, all calculations can be carried out in terms of arbitrary kernels. To this end, we only need to substitute all occurrences of $\mathbf{x} \in \mathbb{R}^d$ in the proofs by feature vectors $\phi(\mathbf{x}) \in \mathcal{H}$. Likewise, all occurrences of inner products $\langle \mathbf{x}, \mathbf{y} \rangle$ generalize to scalar products $\langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$ in the kernel Hilbert space. The scalar product also induces a norm, defined by $\|\mathbf{w}\| := \sqrt{\langle \mathbf{w}, \mathbf{w} \rangle}$ for all $\mathbf{w} \in \mathcal{H}$. The expectation operator $E : \mathcal{H} \mapsto \mathcal{H}$ is well-defined as long as $E\|\mathbf{x}\|$ is finite, which will always be assumed (as a matter of fact, we assume that $\|\mathbf{x}\|$ is bounded almost surely; cf. Axiom 6 and Axiom 10).

It is interesting to discuss whether the assumptions on the distribution generating the innocuous data as imposed by Axiom 6 and Axiom 10 can be fulfilled in Hilbert spaces. For Axiom 6, there is no restriction at all, but in Axiom 10 we use the assertion $P_{(\mathbf{x}-\mathbf{c}_0)\cdot\mathbf{a}} = P_{-(\mathbf{x}-\mathbf{c}_0)\cdot\mathbf{a}}$, which means that the distribution of the data projection onto the attack direction is required to be point symmetric w.r.t. $\mathbf{c}_0 \cdot \mathbf{a}$, where $\mathbf{c}_0$ is the center of the hypersphere before the attack takes place. For example, this

is fulfilled by distributions that are symmetric with respect to $\mathbf{c}_0$ in feature space, and thus naturally fulfilled by Gaussian distributions with mean $\mathbf{c}_0$ (using $\phi = \mathrm{id}$) or truncated Gaussians. When using kernels the symmetry assumption can be invalid, for example, for RBF kernels. However, our empirical analysis (see next section) shows that our bounds are, nevertheless, sharp in practice.

It is also worth noting that the use of kernels can impose geometric constraints on the optimal attack. Note that, in practice, the attacker can only construct attack points in the input space and not directly in the feature space. The attack is then embedded into the feature space. Thus, strictly speaking, we would need to restrict the search space to feature vectors that have a valid pre-image when using kernels. However, this can be a hard problem to solve in general (Fogla and Lee, 2006). In Proposition 11, we do not take this additional complication into account. Therefore, we *over*estimate the attacker. This is admissible for security analysis; it is the *under*estimation of the attack capability that would have been problematic.

8. Case Study: Application to Intrusion Detection

In this section we present an experimental evaluation of the developed analytical instruments in the context of a particular computer security application: intrusion detection. After a short presentation of the application, data collection, preprocessing and model selection, we report on experiments aimed at the verification of the theoretically obtained growth rates for the attack progress as well as the computation of constant factors for specific real-life exploits.[5]

8.1 Anomaly-Based Intrusion Detection

Computer systems linked to the Internet are exposed to a plethora of network attacks and malicious code. Numerous threats, which range from simple "drive-by-downloads" of malicious code to sophisticated self-proliferating worms, target network hosts every day; networked systems are generally at risk to be remotely compromised and abused for illegal purposes. Sometimes it suffices for malware to send a single HTTP-request to a vulnerable webserver to infect a vast majority of computers within minutes (e.g., the Nimda worm). While early attacks were developed rather for fun than for profit, proliferation of current network attacks is now driven by a criminal underground economy. Compromised systems are often abused for monetary gains including the distribution of spam messages and theft of confidential data. The success of these illegal businesses poses a severe threat to the security of network infrastructures. Alarming reports on an expanding dissemination of advanced attacks render sophisticated security systems indispensable (e.g., Microsoft, 2008; Symantec, 2008).

Conventional defenses against such malicious software rest on abuse detection; that is, identifying attacks using known patterns of abuse, so-called attack *signatures*. While abuse detection effectively protects from known threats, it increasingly fails to be able to cope with the amount and diversity of attacks. The time span required for crafting a signature from a newly discovered attack is insufficient for protecting against rapidly propagating malicious code (e.g., Moore et al., 2002; Shannon and Moore, 2004). Moreover, recent attacks frequently use polymorphic modifications, which strongly impedes the creation of accurate signatures (Song et al., 2007). Consequently, there

5. The term *exploit* denotes a sequence of bytes which, given as an input to a vulnerable program, causes execution of arbitrary, potentially harmful code.

is currently a strong demand for alternative techniques for detection of attacks during the course of their propagation.

Anomaly detection methods provide a means for identifying unknown and novel attacks in network traffic and thereby complement regular security defenses. The centroid anomaly detection is especially appealing because of its low computational complexity.[6] It has been successfully used in several well-known intrusion detection systems (e.g., Hofmeyr et al., 1998; Lazarevic et al., 2003; Wang and Stolfo, 2004; Laskov et al., 2004b; Wang et al., 2005, 2006; Rieck and Laskov, 2007).

8.2 Data Corpus and Preprocessing

The data to be used in our case study was collected by recording real HTTP traffic for 10 days at Fraunhofer Institute FIRST. We consider the data at the level of HTTP requests which are the basic syntactic elements of the HTTP protocol. To transform the raw data into HTTP requests, we remove packet headers from the Ethernet, IP and TCP layers, and merge requests spread across multiple packets. After this point, we consider only request bodies (viewed as a byte string) to be our data point. The resulting data set comprises 145,069 requests of the average length of 489 bytes, from which we randomly drew a representative subset of 2950 data points. This data is referred to as the normal data pool.

The malicious data pool is obtained by a similar procedure applied to the network traffic generated by examples of real attacks used for penetration testing.[7] It contains 69 attack instances from 20 real exploits obtained from the Metasploit penetration testing framework.[8] Attacks were launched in a virtual network and normalized to match the characteristics of the innocuous HTTP requests (e.g., URLs were changed to that of a real web server).

As byte sequences are not directly suitable for the application of machine learning algorithms, we deploy a k-gram spectrum kernel (Leslie et al., 2002; Shawe-Taylor and Cristianini, 2004) for the computation of inner products. This kernel represents a linear product in a feature space in which dimensions correspond to subsequences of length k contained in input sequences. To enable fast comparison of large byte sequences (a typical sequence length is 500–1000 bytes), efficient algorithms using sorted arrays (Rieck and Laskov, 2008) were implemented. Furthermore, kernel values are normalized according to

$$k(\mathbf{x},\bar{\mathbf{x}}) \longmapsto \frac{k(\mathbf{x},\bar{\mathbf{x}})}{\sqrt{k(\mathbf{x},\mathbf{x})k(\bar{\mathbf{x}},\bar{\mathbf{x}})}} \tag{19}$$

to avoid a dependence on the length of a request payload. The resulting inner products were subsequently used by an RBF kernel. Notice that if k is a kernel (in our case it is; see Leslie et al., 2002), then the kernel normalized by Equation (19) is a kernel, too. For example, this can be seen by noting that, for any kernel matrix, this normalization preserves its positive definiteness as it is a columnwise operation and thus can only change the principal minors by a constant factor.

8.3 Learning Model

The feature space selected for our experiments depends on two parameters: the k-gram length and the RBF kernel width σ. Prior to the main experiments aimed at the validation of the proposed

6. With suitable parallelization for multicore architectures, processing speeds of over 3 Gbps can be attained.
7. *Penetration testing* refers to launching real exploits against computer systems to identify potential vulnerabilities.
8. Metasploit can be found at `http://www.metasploit.com/`.

security analysis techniques, we investigate optimal model parameters in our feature space. The parameter range considered is $k = 1,2,3$ and $\sigma = 2^{-5}, 2^{-4.5}, ..., 2^{5}$. Each of the 69 attack instances is represented by a feature vector. We refer to these embedded attacks as the *attack points*; that is, the points in the feature space that the adversary would like to have declared as non-anomalous.

To carry out model selection, we randomly partitioned the innocuous corpus into disjoint training, validation and test sets (of sizes 1000, 500 and 500). The training set is used for computing the centroid, the validation set is used for model selection, and the test set is used to evaluate the detection performance of the centroid. The training set is comprised of the innocuous data only, as the online centroid learner assumes clean training data. The validation and test sets are mixed with 10 randomly chosen attack instances. We thereby ensure that none of the attack instances mixed into the validation set has a class label that also occurs in the test data set. When sampling, we realize this requirement by simply skipping instances that would violate this condition.[9] For each partition, an online centroid learner model is trained on a training set and evaluated on a validation and a test set, using the normalized $\text{AUC}_{[0,0.01]}$ (area under the ROC-curve for false positive rates less that 0.01) as a performance measure.[10] For statistical significance, model selection is repeated 1000 times with different randomly drawn partitions. The average values of the normalized $\text{AUC}_{[0,0.01]}$ for the different k values on test partitions are given in Table 2.

It can be seen that the 3-gram model consistently shows better AUC values for both the linear and the best RBF kernels. We have chosen the linear kernel for the remaining experiments since it allows one to carry out computations directly in the input space with only a marginal penalty in detection accuracy.

	linear	best RBF kernel	optimal σ
1-grams	0.913 ± 0.051	0.985 ± 0.021	$2^{-2.5}$
2-grams	0.979 ± 0.026	0.985 ± 0.025	$2^{-1.5}$
3-grams	0.987 ± 0.018	0.989 ± 0.017	$2^{-0.5}$

Table 2: AUC for the linear kernel, the best RBF kernel and the optimal bandwidth σ.

8.4 Intrinsic HTTP Data Dimensionality

As it was shown in Sec. 4.2.3, the dimensionality of the training data plays a crucial role in the (in)security of the online centroid learner when using the nearest-out update rule. In contrast, the displacement under the average-out rule is independent of the input dimensionality; we therefore focus on the nearest-out rule in this section. For the intrusion detection application at hand, the dimensionality of the chosen feature space (k-grams with $k = 3$) is 256^3, that is, it is rather high. One would thus expect a dramatic impact of the dimensionality on the displacement (an thus the insecurity) of the learner. However, the real progress rate depends on the *intrinsic* dimensionality of the data. When the latter is smaller than the size of the training data, an attacker can compute a PCA of the data matrix (Schölkopf et al., 1998) and project the original data into the subspace spanned by a smaller number of informative components. The following theorem shows that the dimensionality of the relevant subspace in which attack takes place is bounded by the size of the

9. The latter requirement reflects the goal of anomaly detection to recognize *previously unknown* attacks.
10. The normalization is such that AUC = 1 holds for a perfect detector.

training data n, which can be much smaller than the input dimensionality, typically in the range of 100–1000 for realistic applications.

Theorem 16 *There exists an optimal solution of problem (11) satisfying*

$$\mathbf{x}_i^* \in \text{span}(\mathbf{a}, \mathbf{x}_1, \ldots, \mathbf{x}_n) .$$

The above theorem, which also can be used as a representer theorem for "kernelization" of the optimal greedy attack, shows that the attack's efficiency cannot be increased beyond dimensions with $d \geq n+1$. The proof is given in Appendix A.1.

To determine the intrinsic dimensionality of possible training sets drawn from HTTP traffic, we randomly draw 1000 elements from the innocuous pool, calculate a linear kernel matrix in the space of 3-grams and compute its eigenvalue decomposition. We then determine the number of leading eigen-components as a function of the percentage of variance preserved. The results averaged over 100 repetitions are shown in Figure 8.

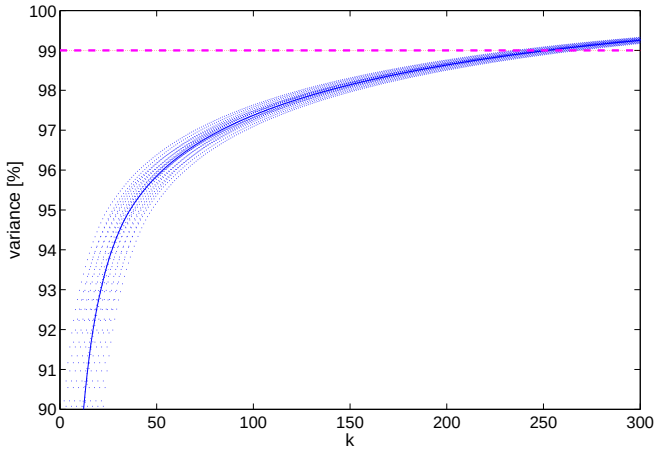


Figure 8: Intrinsic dimensionality of the embedded HTTP data. The preserved variance is plotted as a function of the number of eigencomponents, k, employed for calculation of variance (solid blue line). The tube indicates standard deviations.

It can be seen that 250 kernel PCA components are needed to preserve 99% of the variance. This implies that, although the effective dimensionality of the HTTP traffic is significantly smaller than the number of training data points, it still remains sufficiently high so that the attack progress rate approaches 1, which is similar to the simple average-out learner.

8.5 Geometrical Constraints of HTTP Data

Several technical difficulties arising from data geometry have to be overcome in launching a poisoning attack in practice. However, consideration of the training data geometry provides an attacker with efficient tools for finding reasonable approximations for the above mentioned tasks.

(1) Even for the linear kernel, it is hard to craft a poisoning point in the 3-gram input space due to the high dimensionality of the 3-gram space. An approximately equivalent explicit feature space can be constructed by applying kernel PCA to the kernel matrix K. By pruning the eigenvalues "responsible" for dimensions with low variance one can reduce the size of the feature space to the implicit dimensionality of the problem if the kernel matches the data (Braun et al., 2008). In all subsequent experiments we used $d = 256$ as suggested by the experiments in Section 8.4.

(2) The normalization condition (19) requires that a solution lies on a unit sphere.[11] Unfortunately, this renders the calculation of the greedy-optimal attack point non-convex. Therefore, we pursue the following heuristic procedure to enforce normalization: we explicitly project local solutions (for each Voronoi cell) to a unit sphere, verify their feasibility (the radius and the cell constraints) and remove infeasible points from the outer loop (10).

(3) In general one cannot expect each feature space vector to correspond to a valid byte sequence since not all combinations of k-grams can be "glued" to a valid byte sequence. In fact, finding a sequence with the best approximation to a given k-gram feature vector is NP-hard (Fogla and Lee, 2006). Since the optimal attack lies in the span of the training data (cf. Theorem 16) we can construct an attack byte sequence by concatenating original training sequences with rational coefficients that approximately match the coefficients of a linear combination. A potential disadvantage of this method is the increase of sequence lengths. Large requests are conspicuous and may consume significant resources on the attacker's part.

(4) An attack byte sequence must be embedded in a valid HTML protocol frame so that a request does not cause an error on a server. An HTTP request consists of fixed format headers and a variable format body. A straightforward way to stealthily introduce arbitrary content is to provide a body in a request whose method (e.g., GET) does not require one (it is ignored by the server). Alternatively one can introduce custom headers that are not expected by the server and will be ignored as well.

8.6 Poisoning Attack for Finite Horizon Centroid Learner

The analysis carried out in Section 4 shows that an online centroid learner, in general, does not provide sufficient security if an attacker fully controls the data. Practical efficiency of a poisoning attack, however, depends on the dimensionality and geometry of the training data analyzed in the previous section. Theoretical results were validated in simulations on artificial data presented in Section 4.2.3. The experiments in this section are intended to verify the results presented in Section 4.2.3 in the context of real attacks against HTTP applications. Our experiments focus on the nearest-out learner as other update rules can be easily attacked with trivial methods.

The experimental protocol is as follows. We randomly draw $n = 250$ training points from the innocuous corpus, calculate the center of mass and fix the radius such that the false positive rate on the training data is $\alpha = 0.001$. Then we draw a random instance from each of the 20 attack classes and for each of these 20 attack instances generate a poisoning attack as described in Section 8.5. An attack succeeds when the attack point is accepted as innocuous by the learning algorithm.

For each attack instance, the number of iterations needed for an attack to succeed and the respective displacement of the center of mass is recorded. Figure 9 shows, for each attack instance, the behavior of the relative displacement at the point of success as a function of the number of iterations. We interpolate a "displacement curve" from these pointwise values by linear least-squares

11. In the absence of normalization, the high variability of the byte sequence lengths leads to poor accuracy of centroid anomaly detection.

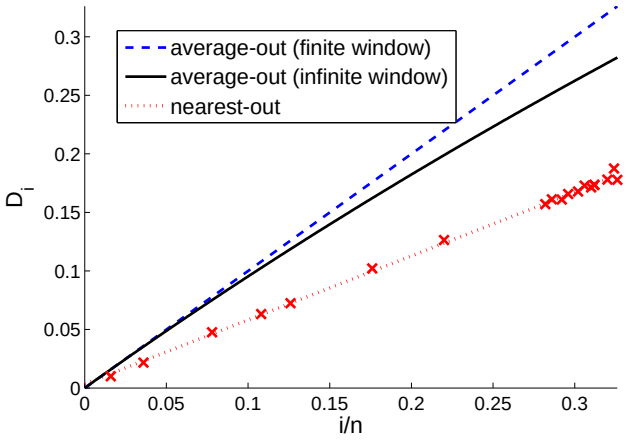


Figure 9: Empirical displacement of the nearest-out centroid for 20 different exploits (crosses, linear fit shown by a red dotted line). Displacement values are shown at the point of success for each attack. Theoretical bounds are shown for comparison (blue and black lines).

regression. For comparison, the theoretical upper bounds for the average-out and all-in cases are shown ("all-in" hereby refers to the update strategy given by Equation (1), where points are added to the training set without removing old ones). Notice that the bound for the all-in strategy is also almost linear for the small i/n ratios observed in this experiment.

The observed results *confirm that the linear progress rate in the full control scenario can be attained for real data*. Compared to the simulations of Section 8.5, the progress rate of an attack is approximately half the one for the average-out case. This can be attributed to multiple approximations performed in the attack implementation for real byte sequences. For example, we use a k-gram spectrum kernel, so each poisoning attack point is restricted to have unit norm in feature space. The practicality of the poisoning attack is further emphasized by the small number of iterations needed for an attack to succeed: it suffices to overwrite between 2 and 35 percent of the initial number of points in the training data to subvert the nearest-out learner.

8.7 Critical Traffic Ratios of HTTP Attacks

For the case of the attacker's limited control over the data, the success of a poisoning attack largely depends on attacker's constraints, as shown in the analysis in Sections 5 and 6. The main goal of the experiments in this section is therefore to investigate the impact of potential constraints in practice. In particular, we are interested in the impact of the traffic ratio ν and the false positive rate α.

The analysis in Section 5 (cf. Theorem 8 and Figure 5) shows that the displacement of a poisoning attack is bounded from above by a constant depending on the traffic ratio ν controlled by an attacker. Hence the susceptibility of the learner to a particular attack depends on the *value of this constant*. If an attacker does not control a sufficiently large traffic portion and the potential displacement is bounded by a constant smaller than the distance from the initial center of mass to

the attack point, then the attack fails. To illustrate this observation, we compute critical traffic rates needed for the success of attacks from each of the 20 attack classes in our malicious pool.

We randomly draw a 1000-element training set from the innocuous pool and calculate its center of mass (in the space of 3-grams). The radius is fixed such that the false positive rate $\alpha = 0.001$ on innocuous data is attained. For each of the 20 attack classes we compute the class-wise median distance to the centroid's boundary. Using these distance values we calculate the "critical value" ν_{crit} by solving Theorem 8(c) for ν (cf. Equation (18)). The experiments were repeated 10 times, with results shown in Table 3.

Attacks	**Rel. dist.**	ν_{crit}
ALT-N WebAdmin Overflow	0.058 ± 0.002	0.055 ± 0.002
ApacheChunkedEncoding	0.176 ± 0.002	0.150 ± 0.001
AWStats ConfigDir Execution	0.067 ± 0.002	0.063 ± 0.002
Badblue Ext Overflow	0.168 ± 0.002	0.144 ± 0.001
Barracuda Image Execution	0.073 ± 0.002	0.068 ± 0.002
Edirectory Host	0.153 ± 0.002	0.132 ± 0.001
IAWebmail	0.178 ± 0.002	0.151 ± 0.001
IIS 5.0 IDQ exploit	0.162 ± 0.002	0.140 ± 0.001
Pajax Execute	0.107 ± 0.002	0.097 ± 0.002
PEERCAST URL	0.163 ± 0.002	0.140 ± 0.001
PHP Include	0.097 ± 0.002	0.088 ± 0.002
PHP vBulletin	0.176 ± 0.002	0.150 ± 0.001
PHP XML RPC	0.172 ± 0.002	0.147 ± 0.001
HTTP tunnel	0.160 ± 0.002	0.138 ± 0.001
IIS 4.0 HTR exploit	0.176 ± 0.002	0.149 ± 0.002
IIS 5.0 printer exploit	0.161 ± 0.002	0.138 ± 0.001
IIS unicode attack	0.153 ± 0.002	0.133 ± 0.001
IIS w3who exploit	0.168 ± 0.002	0.144 ± 0.001
IIS 5.0 WebDAV exploit	0.179 ± 0.002	0.152 ± 0.001
rproxy exploit	0.155 ± 0.002	0.134 ± 0.001

Table 3: Relative distances (in radii) of exploits to the boundary of a centroid enclosing all training points and critical values of parameter ν.

The results indicate that in order to subvert an online centroid learner an attacker needs to control on average from 5 to 20 percent of traffic (with small variance). This could be a significant limitation on highly visible sites. Generating sufficiently high bandwidths in this case is likely to make the attacker's cost exorbitantly high.

On the other hand, one can see that the traffic rate limiting alone cannot be seen as a sufficient protection instrument due to its passive nature. In the following section we investigate a different protection scheme using both the traffic ratio and the false positive rate control.

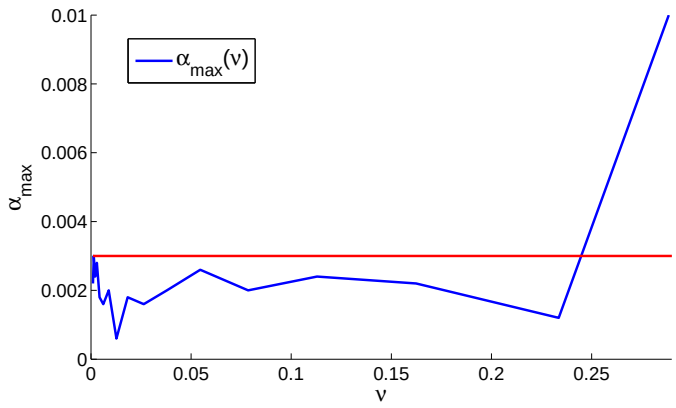


Figure 10: Maximal false positive rate within 10000 attack iterations as a function of ν (maximum taken over 10 runs).

8.8 Poisoning Attack Against Learner with False Positive Protection

The analysis in Section 5 (cf. Theorem 8 and Figure 5) shows that the displacement of a poisoning attack is bounded from above by a number, depending on the traffic ratio ν and the maximal false positive rate α. The dependence of this number on α can be used for *constructive* protection of a learner against a poisoning attack. The idea is to use the observed false positive rate as a measure of the attack's progress, and to turn online updates off if some critical false alarm rate is attained.

8.8.1 EXPERIMENT 1: PRACTICABILITY OF FALSE POSITIVE PROTECTION

The first issue to be decided is what values of α are reasonable from the operational perspective. A false positive threshold that is too low would lead to a quasi-permanent shutdown of online updates; a high tolerance to false positives would allow an attack to slip in unnoticed. Hence, we need to investigate the dependence of the observed false positive rate on the traffic ratio ν under the attacker's control.

To this end, we randomly draw 100 samples from the innocuous pool (each sample containing 1000 points), and compute the mean and the radius that encompasses 99.9% of the points. We may hence view the average radius r (averaged over the 100 samples) as the empirical estimate of the $\alpha = 0.001$-quantile of the innocuous pool. Then, we randomly draw another 1000-element training set from the innocuous pool and use it to calculate the center $\mathbf{c}$. We thus may expect the resulting initial centroid $(\mathbf{c}, r)$ to have a false positive rate of $\alpha = 0.001$ on the innocuous pool. Next, the learner is switched online. We randomly draw a 500-element online training set and a 500-element *hold-out set* from the innocuous pool. The hold-out set is used for the estimation of the false positive rate for each location of the center. We then proceed by presenting normal data (drawn with replacement from the online training set) mixed with poisoning attack points (using the IIS 5.0 WebDAV exploit as target) and measuring the false positive rate for each attack iteration. Notice that normal data points may also get rejected if they do not fall within the radius r from the (poisoned) center.

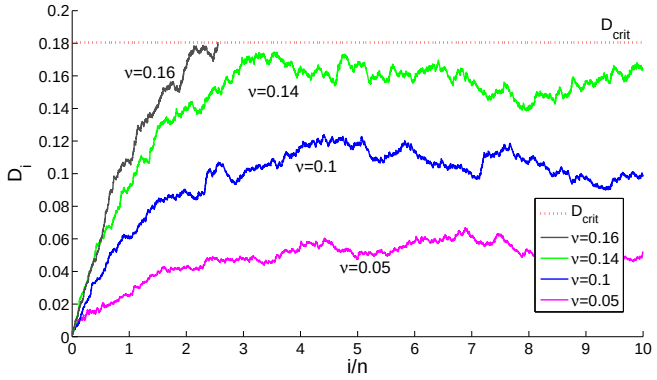


Figure 11: Simulation of a poisoning attack against *IIS 5.0 WebDAV exploit* under limited control.

In Figure 10 the maximal observed false positive rate is shown for various values of ν, where the maximum is taken over all attack iterations and 10 runs. One can see from the plot that $\alpha = 0.005$ is a reasonable threshold in our setting to ensure that the system's normal operation is not disrupted by the false positive protection.

8.8.2 EXPERIMENT 2: ATTACK SIMULATION FOR FALSE POSITIVE PROTECTION

In the previous experiment we have seen that $\alpha = 0.005$ is a reasonable threshold for the false positive protection. In this section we illustrate that the critical values from Section 8.7 computed on the basis of Theorem 8 for the maximal false positive rate of $\alpha = 0.005$ give a good approximation of the true impact of a poisoning attack.

To this end, we fix a particular attack in our malicious corpus (IIS WebDAV 5.0 exploit) and run a poisoning attack against the average-out centroid learner for various values of $\nu \in [0.05, 0.10, 0.14, 0.16]$ recording the actual displacement curves. One can see from Figure 11 that the attack succeeds for $\nu = 0.16$ but fails to reach the required relative displacement of $D_{crit} = 0.18$ for $\nu = 0.14$. The theoretical critical traffic ratio for this attack (with false positive rate bounded by $\alpha \leq 0.005$) according to Table 3 is $\nu_{\text{crit}} = 0.152$. The experiment shows that the derived bounds are surprisingly tight in practice.

The IIS WebDAV 5.0 exploit is, in a sense, an extreme case as it constitutes the most pessimistic scenario for an attacker (farthest away from the normality centroid). In another experiment, we therefore consider the ALT-N WebAdmin Overflow; that is, the most *optimistic* scenario for the attacker and the closest attack to the centroid. The result is shown in Figure 12. We observe from the figure that the experiment again supports our theory: the predicted critical fraction is $\nu = 0.055$ and indeed this quickly leads to a successful attack. For a slightly smaller $\nu = 0.45$, the attack fails and the IDS can be considered safe.

8.8.3 IMPLEMENTATION OF A POISONING PROTECTION IN PRACTICE

In Section 5, we have seen that an attacker's impact on corrupting the training data crucially depends on the fraction of adversarial points in the training data stream. This implies that a large amount of innocuous training points must be processed in order for the system to be secure. In Section 6, we have seen that we can secure the learner by setting a threshold on the false positive rate α. When a false positive rate exceeds the threshold, further countermeasures such as disabling the online

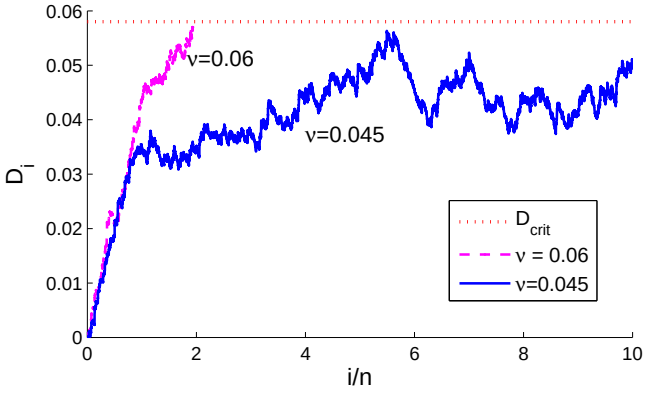


Figure 12: Simulation of a poisoning attack against *ALT-N WebAdmin Overflow* under limited control.

training process can be triggered. This raises an issue of a reliable estimation of the allowable false positive rate.

In practice, this can be done; for example, by caching the training data. When the cache exceeds a certain value at which we have a confident estimation of α (e.g., after 24 hours), the cached training data can be applied to the learner. Since in applications including intrusion detection we usually deal with a very large amount of training data, a confident estimation is already possible after a short time period.

9. Discussion and Conclusions

Understanding of security properties of learning algorithms is essential for their protection against abuse. In order to prove the immunity of a learning algorithm against manipulated data attacks, certain security properties must be *proved*. To this end, we have developed a methodology for security analysis of learning algorithms and applied it for a specific scenario of online centroid anomaly detection. Our analysis highlights conditions under which an attacker's effort to subvert this algorithm is prohibitively high. We further propose some constructive countermeasures for protecting online anomaly detection against poisoning attacks.

Our work is related to other research in machine learning for adversarial environments that has gained significant attention in recent years. A comprehensive survey of a large body of work in this field can be found in Barreno et al. (2010). The majority of related work targets classification problems (e.g., Kearns and Li, 1993; Dalvi et al., 2004; Globerson and Roweis, 2006; Dekel and Shamir, 2008; Dekel et al., 2009). In contrast, our work is focused on anomaly detection methods that have received much less previous attention. Compared to the closest related work of Nelson and Joseph (2006), our methods address more realistic attack and learning scenarios. In particular, we dispense with the assumption of an infinite amount of training data accumulated in the course of training and analyze more advanced scenarios in which the attacker's impact is limited by certain constraints.

Other related work for anomaly detection methods is concerned with the PAYL (Wang and Stolfo, 2004) and Anagram (Wang et al., 2006) algorithms. Both can be seen as a special case of centroid anomaly detection with appropriate distance functions. The blending attack considered by

Fogla et al. (2006); Fogla and Lee (2006) is aimed at evading a trained model at the detection stage by modifying malicious data to resemble the normal one. It differs from the scenario considered in our work in that the attack takes place *after* the model has been trained, whereas the poisoning attack affects the model in the course of training. Although the exact blending was shown to be NP-hard for k-gram feature spaces, approximations have been proposed that are good enough in practice, at least for low values of k. Another related technique is sanitization proposed by Cretu et al. (2008). It deals with the general problem of cleaning the training data from potential attacks. It was shown that small amounts of unintentional attacks can be filtered out by building micro-models based on parts of the training data and rejecting data deemed as anomalous by some micro-models. This work differs from ours in that it considers offline learning; that is, the sanitization takes place *before* before the training data is presented to the learning algorithm. An extension of sanitization to online learning scenarios has been proposed in Cretu-Ciocarlie et al. (2009). The extended method uses previously constructed micro-models to sanitize data from which the most recent model is learned. However, once the attacker has knowledge about individual micro-models, a poisoning attack similar the one considered in our work can be constructed. If suffices for an attacker to craft points that are accepted by all micro-models and inject them into the training data to poison a new micro-model.

Some of the previously developed methods can also be seen to contain parts of the general analysis methodology proposed in Section 1.1. Dalvi et al. (2004) analyzed the robustness of Bayesian classification against adversarial impact. The choice of their classifier is motivated by widespread application of the naive Bayes classification in the domain of spam detection where real examples of adversarial impact have long been observed. The adversarial classification is considered as a game between an attacker and a learner. Due to the complexity of analysis, only one move by each party can be analyzed. Similar to our approach, Dalvi et al. (2004) formalize the problem by defining cost functions of an attacker and a learner (Step 1) and determine an optimal adversarial strategy (Step 3). Although the attacker's constraints are not explicitly treated theoretically, several scenarios using specific constraints have been tested experimentally. No analysis of the attacker's gain is carried out; instead, the learner's direct response to adversarial impact is considered.

A somewhat related approach has been developed for handling worst-case random noise; for example, random feature deletion (Globerson and Roweis, 2006; Dekel and Shamir, 2008). Similar to Dalvi et al. (2004), both of these methods construct a classifier that automatically reacts to the worst-case noise or, equivalently, the optimal adversarial strategy. In both methods, the learning problem is formulated as large-margin classification using a specially constructed risk function. An important role in this approach is played by the consideration of constraints (Step 2); for example, in the form of the maximal number of corruptible features. Although these approaches do not quantitatively analyze the attacker's gain, Dekel and Shamir (2008) contains an interesting learning-theoretic argument that relates classification accuracy and sparseness with the robustness against adversarial noise.

To summarize, we believe that despite recent evidence of possible attacks against machine learning and the currently lacking theoretical foundations for learning under adversarial impact, machine learning algorithms *can* be protected against such an impact. The key to such a protection lies in quantitative analysis of the security of machine learning. We have shown that such an analysis can be rigorously carried out for specific algorithms and attacks. Further work should extend this analysis to more complex learning algorithms and a wider attack spectrum.

Acknowledgments

The authors greatly thank Peter Bartlett, Ulf Brefeld, Vojtech Franc, and Klaus-Robert Müller, Konrad Rieck for fruitful discussions and helpful comments. Furthermore we thank Konrad Rieck for providing the network traffic and an anonymous reviewer for many helpful comments. Most of the work was done when MK and PL were supported by the German Bundesministerium für Bildung und Forschung (BMBF) under the project REMIND (FKZ 01-IS07007A), and by the FP7-ICT Programme of the European Community, under the PASCAL2 Network of Excellence, ICT-216886. MK achknowledges a postdoctorial fellowship and PL a Heisenberg fellowship, both by the National Research Foundation of Germany (DFG). MK is also supported by the German Science Foundation under DFG MU 987/6-1, RA 1894/1-1, and by the World Class University Program through the National Research Foundation of Korea funded by the Korean Ministry of Education, Science, and Technology, under Grant R31-10008.

Appendix A. Auxiliary Material and Proofs

In this appendix we present some helpful Lemmas as well as detailed proofs of the theorems.

A.1 Auxiliary Material for Section 4

This appendix start with a characterization of the greedy-optimal optimization problem in terms of dual variables.

A.1.1 Proof of Theorem 16

Proof The Lagrangian of optimization problem (11) is given by:

$$\begin{aligned} L(\mathbf{x},\alpha,\beta) &= -(\mathbf{x}-\mathbf{x}_i)\cdot\mathbf{a}+\sum_{j=1}^{n}\alpha_j\left(2(\mathbf{x}_j-\mathbf{x}_i)\cdot\mathbf{x}-\mathbf{x}_j\cdot\mathbf{x}_j+\mathbf{x}_i\cdot\mathbf{x}_i\right) \\ &\quad +\beta\left(\mathbf{x}\cdot\mathbf{x}-\frac{2}{n}\sum_{j=1}^{n}\mathbf{x}\cdot\mathbf{x}_j+\frac{1}{n^2}\sum_{j,k=1}^{n}\mathbf{x}_j\cdot\mathbf{x}_k-r^2\right). \end{aligned}$$

Since the feasible set of problem (11) is bounded by the spherical constraint and is not empty ($\mathbf{x}_i$ trivially is contained in the feasible set), there exists at least one optimal solution $\mathbf{x}_i^*$ to the primal. For optimal $\mathbf{x}_i^*$, α^* and β^*, we have the following first order optimality conditions

$$\frac{\delta L}{\delta \mathbf{x}}=0: \quad -\mathbf{a}-\frac{1}{n}\sum_{j=1}^{n}\mathbf{x}_j+2\sum_{j=1}^{n}\alpha_j^*(\mathbf{x}_j-\mathbf{x}_i)+\beta^*\left(2\mathbf{x}_i^*-\frac{2}{n}\sum_{j=1}^{n}\mathbf{x}_j\right)=0. \tag{20}$$

If $\beta^*\neq 0$, the latter equation can be resolved for $\mathbf{x}_i^*$ leading to:

$$\mathbf{x}_i^*=\frac{1}{2\beta^*}\mathbf{a}+\sum_{j=1}^{n}\left(\frac{1}{2\beta^* n}-\frac{\alpha_j^*}{\beta^*}+\frac{1}{n}\right)\mathbf{x}_j+\frac{1}{\beta^*}\sum_{j=1}^{n}\alpha_j^*\mathbf{x}_i.$$

From the latter equation, we see that $\mathbf{x}$ is contained in $S:=\operatorname{span}(\mathbf{x}_1,...,\mathbf{x}_n \text{ and } \mathbf{a})$.

Now assume $\beta^*=0$ and $\mathbf{x}_i^*\notin S$. At first, since $\beta^*=0$, we see from Equation (20) that $\mathbf{a}$ is contained in the subspace $S:=\operatorname{span}(\mathbf{x}_1,...,\mathbf{x}_n)$. Hence the objective, $(\mathbf{x}-\mathbf{x}_i)\cdot\mathbf{a}$, only depends on the

optimal $\mathbf{x}$ via inner products with the data $\mathbf{x}_i$. The same naturally holds for the constraints. Hence, both the objective value and the constraints are invariant under the projection of $\mathbf{x}_i^*$ onto S, denoted by P. Hence $P(x_i^*)$ also is an optimal point. Moreover, by construction, $P(\mathbf{x}_i^*) \in S = \text{span}(\mathbf{x}_1^*, ..., \mathbf{x}_n^*)$. ■

A.1.2 THEORETICAL ANALYSIS FOR THE OPTIMAL GREEDY ATTACK

The dependence of an attack's effectiveness on data dimensionality results from the geometry of Voronoi cells. Intuitively, the displacement at a single iteration depends on the size of the largest Voronoi cell in a current working set. Although it is hard to derive a precise estimate on the latter, the following "average-case" argument sheds some light on the attack's behavior, especially since it is the average-case geometry of the working set that determines the overall attack progress.

Consider a simplified case where each of the Voronoi cells C_j constitutes a ball of radius r centered at a data point $\mathbf{x}_j$, $j = 1, \ldots, n$. Clearly, the greedy attack will result in a progress of r/n (we will move one of the points by r but the center's displacement will be discounted by $1/n$). We will now use the relationship between the volumes of balls in $\mathbb{R}^d$ to relate r, R and d.

The volume of each Voronoi cell C_j is given by

$$\text{Vol}(C_j) = \frac{\pi^{\frac{d}{2}} r^d}{\Gamma\left(\frac{d}{2}+1\right)}.$$

Likewise, the volume of the hypersphere S of radius R is

$$\text{Vol}(S) = \frac{\pi^{\frac{d}{2}} R^d}{\Gamma\left(\frac{d}{2}+1\right)}.$$

Assuming that the Voronoi cells are "tightly packed" in S, we obtain

$$\text{Vol}(S) \approx n\,\text{Vol}(C_j).$$

Hence we conclude that

$$r \approx \sqrt[d]{\frac{1}{n}}\, R.$$

One can see that the attacker's gain approximately represented by the cell radius r is a constant fraction of the threshold R, which explains the linear progress of the poisoning attack. The slope of this linear dependence is controlled by two opposing factors: the size of the training data decreases the attack speed whereas the intrinsic dimensionality of the feature space increases it. Both factors depend on fixed parameters of the learning problem and cannot be controlled by the algorithm. In the limit, when d approaches n (the effective dimension is limited by the training data set according to Theorem 16) the attack progress rate is approximately described by the function $\sqrt[n]{\frac{1}{n}}$ which approaches 1 with increasing n.

A.2 Proofs of Section 5

This following proposition is a well-known fact from algebra.

Proposition 17 (Geometric series) *Let $(s)_{i\in\mathbb{N}_0}$ be a sequence of real numbers satisfying $s_0 = 0$ and $s_{i+1} = qs_i + p$ (or $s_{i+1} \leq qs_i + p$ or $s_{i+1} \geq qs_i + p$) for some $p, q > 0$. Then*

$$s_i = p\frac{1-q^i}{1-q}\,, \quad \left(\text{and } s_i \leq p\frac{1-q^i}{1-q} \text{ or } s_i \geq p\frac{1-q^i}{1-q}\right), \tag{21}$$

respectively.

Proof We prove part (a) of the theorem by induction over $i \in \mathbb{N}_0$, the case of $i = 0$ being obvious.

In the inductive step we show that if Equation (21) holds for an arbitrary fixed i it also holds for $i+1$:

$$\begin{aligned} s_{i+1} &= qs_i + p = q\left(p\frac{1-q^i}{1-q}\right) + p = p\left(q\frac{1-q^i}{1-q} + 1\right) \\ &= p\left(\frac{q - q^{i+1} + 1 - q}{1-q}\right) = p\left(\frac{1-q^{i+1}}{1-q}\right). \end{aligned}$$

The proof of part (b) is analogous. ■

Proof of Theorem 8(b) Multiplying both sides of Equation (17) with $\mathbf{a}$ and substituting $D_i = \mathbf{c}_i \cdot \mathbf{a}$ results in

$$D_{i+1} = \left(1 - \frac{1-B_i}{n}\right) D_i + \frac{B_i}{n} + \frac{(1-B_i)}{n}\mathbf{x}_i \cdot \mathbf{a}\,.$$

Inserting $B_i^2 = B_i$ and $B_i(1-B_i) = 0$, which holds because B_i is Bernoulli, into the latter equation, we have:

$$D_{i+1}^2 = \left(1 - 2\frac{1-B_i}{n} + \frac{1-B_i}{n^2}\right) D_i^2 + \frac{B_i}{n^2} + \frac{(1-B_i)}{n^2}\|x_i\|^2 + 2\frac{B_i}{n}D_i + 2\left(1 - \frac{1}{n}\right)\frac{(1-B_i)}{n}D_i x_i\,,$$

where $x_i := \mathbf{x}_i \cdot \mathbf{a}$. Taking the expectation on the latter equation, and noting that, by Axiom 6, $\mathbf{x}_i$ and $\mathbf{D}_i$ are independent, we have:

$$\begin{aligned} E\left(D_{i+1}^2\right) &= \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)\right) E\left(D_i^2\right) + 2\frac{\nu}{n}E(D_i) + \frac{\nu}{n^2} + \frac{1-\nu}{n^2}E(\|\mathbf{x}_i \cdot \mathbf{a}\|^2) \\ &\overset{(1)}{\leq} \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)\right) E\left(D_i^2\right) + 2\frac{\nu}{n}E(D_i) + \frac{1}{n^2}\,, \end{aligned} \tag{22}$$

where (1) holds because, by Axiom 6, we have $\|\mathbf{x}_i\|^2 \leq r$ and moreover $\|\mathbf{a}\| = r$, $r = 1$. Inserting the result of Theorem 8(a) in the latter equation results in the following recursive formula:

$$E\left(D_{i+1}^2\right) \leq \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)\right) E\left(D_i^2\right) + 2(1-a_i)\frac{\nu}{n}\frac{\nu}{1-\nu} + \frac{1}{n^2}\,.$$

By the formula of the geometric series, that is, by Proposition 17, we have:

$$E\left(D_i^2\right) \leq \left(2(1-a_i)\frac{\nu}{n}\frac{\nu}{1-\nu} + \frac{1}{n^2}\right)\frac{1-b_i}{\frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)}\,,$$

where $b_i := \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)\right)^i$. Furthermore, by some algebra

$$E\left(D_i^2\right) \leq \frac{(1-a_i)(1-b_i)}{1-\frac{1}{2n}} \frac{\nu^2}{(1-\nu)^2} + \frac{1-b_i}{(2n-1)(1-\nu)}\,. \tag{23}$$

We will need the auxiliary formula

$$\frac{(1-a_i)(1-b_i)}{1-\frac{1}{2n}} - (1-a_i)^2 \leq \frac{1}{2n-1} + a_i - b_i\,, \tag{24}$$

which can be verified by some more algebra and employing $b_i < a_i$ and that a_i ranges in the real interval $[0,1]$. We finally conclude

$$\begin{aligned}
\mathrm{Var}(D_i) &= E(D_i^2) - (E(D_i))^2 \\
&\overset{\text{Th.12}(a);\ \text{Equation(23)}}{\leq} \left(\frac{(1-a_i)(1-b_i)}{1-\frac{1}{2n}} - (1-a_i)^2\right)\left(\frac{\nu}{1-\nu}\right)^2 + \frac{1-b_i}{(2n-1)(1-\nu)^2} \\
&\overset{\text{Equation(24)}}{\leq} \gamma_i \left(\frac{\nu}{1-\nu}\right)^2 + \delta_n
\end{aligned}$$

where we denote $\gamma_i := a_i - b_i$ and $\delta_n := \frac{\nu^2 + (1-b_i)}{(2n-1)(1-\nu)^2}$, and use $(1-\nu)^2 \leq 1-\nu \leq 1$ in the inequalities. This completes the proof.

■

A.3 Proofs of Section 6

In this appendix we prove one of our main results.

Lemma 18 *Let $\mathcal{C}$ be a online centroid learner with maximal false positive rate α satisfying the optimal attack strategy. Denote $x_i := \mathbf{x}_i \cdot \mathbf{a}$. Then we have:*

$$\begin{aligned}
&\text{(a)} \quad 0 \leq E\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i^q\right) \leq \alpha E(D_i^q)\,, \quad q = 1,2 \\
&\text{(b)} \quad 0 \leq E\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} x_i\right) \leq \alpha \\
&\text{(c)} \quad E\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} x_i D_i\right) \leq \alpha E(D_i)\,.
\end{aligned}$$

Proof

(a) Let $q = 1$ or $q = 2$. Since D_i is independent of $\mathbf{x}_i$ (hence constant under the operator $E_{\mathbf{x}_i}$), we have

$$E_{\mathbf{x}_i}\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i^q\right) = (D_i)^q E_{\mathbf{x}_i}\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}}\right)\,.$$

Hence by Axiom 10

$$E_{\mathbf{x}_i}\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i^q\right) = 0\,,$$

if $E_{\mathbf{x}_i}\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}}\right) > \alpha$ and $0 \leq E_{\mathbf{x}_i}\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}}\right) \leq \alpha$ otherwise. Moreover, $E(D_i) \geq 0$ by the symmetry assumption in Axiom 10. Taking the full expectation $E = E_{\mathbf{c}_i} E_{\mathbf{x}_i}$ yields the assertion (a).

(b) We denote $I_{\leq} := I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}}$ and $I_{>} := I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}}$. Since

$$E(I_{\leq} x_i) + E(I_{>} x_i) = E\left((I_{\leq} + I_{>}) x_i\right) = E(x_i) = 0\,,$$

we conclude

$$E(I_{\leq} x_i) = -E(I_{>} x_i) = E(I_{>}(-x_i)) \overset{(1)}{\leq} \alpha\,,$$

where (1) holds because $||x_i|| \leq 1$ and by Axiom 10 we have $E(I_{>}) \leq \alpha$.

Furthermore $E(I_{\leq} x_i) \geq 0$ is clear.

(c) The proof of (c) is analogous to that of (a) and (b). ■

Proof of Theorem 12

(a) By Axiom 10, we have

$$D_{i+1} \leq \max\left(0, D_i + \frac{1}{n}\left(B_i\left(f(\mathbf{c}_i) - \mathbf{c}_i\right) + (1 - B_i) I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}}\left(\mathbf{x}_i - \mathbf{c}_i\right)\right) \cdot \mathbf{a}\right). \tag{25}$$

By Proposition 11 an greedy-optimal attack strategy can be defined by

$$f(\mathbf{x}) = \mathbf{x} + \mathbf{a}\,.$$

Inserting the latter into Equation (25), using $D_i \overset{\text{Def.}}{=} \mathbf{c}_i \cdot \mathbf{a}$, and taking the expectation, we have

$$E(D_{i+1}) \leq E\left[I_{\left\{E_{\mathbf{x}}\left(I_{\{\|\mathbf{x} - \mathbf{c}_i\| \leq r\}}\right) \leq 1 - \alpha\right\}}\left(D_i + \frac{1}{n}\left(B_i + (1 - B_i) I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}}\left(x_i - D_i\right)\right)\right)\right], \tag{26}$$

where $x_i = \mathbf{x}_i \cdot \mathbf{a}$. By the symmetry assumption in Axiom 10, the first term can be omitted, hence the above equation can be rewritten as

$$\begin{aligned} E(D_{i+1}) \quad &\leq \quad \left(1 - \frac{1 - \nu}{n}\right) E(D_i) + \frac{\nu}{n} \\ &\quad + \frac{1 - \nu}{n}\left(E\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i\right) + E\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} x_i\right)\right). \end{aligned}$$

Inserting the inequalities (a) and (b) of Lemma 18 into the above equation results in:

$$\begin{aligned} E(D_{i+1}) \quad &\leq \quad \left(1 - \frac{1 - \nu}{n}\right) E(D_i) + \frac{\nu}{n} + \frac{1 - \nu}{n}\left(\alpha E(D_i) + \alpha\right) \\ &= \quad \left(1 - \frac{(1 - \nu)(1 - \alpha)}{n}\right) E(D_i) + \frac{\nu + \alpha(1 - \nu)}{n}\,. \end{aligned}$$

By the formula of the geometric series, that is, Proposition 17, we have

$$E(D_{i+1}) \leq (1 - a_i) \frac{\nu + \alpha(1 - \nu)}{(1 - \nu)(1 - \alpha)}\,,$$

where $a_i = \left(1 - \frac{(1-\nu)(1-\alpha)}{n}\right)^i$. Moreover we have

$$E(D_{i+1}) \geq (1 - b_i) \frac{\nu}{1 - \nu}\,, \tag{27}$$

where $b_i = \left(1 - \frac{1-\nu}{n}\right)^i$, by analogous reasoning. We schematically show this by starting from Equation (26) and subsequently applying Jensen's inequality and using the lower bounds of Lemma 18 and the formula for the geometric series. Since $b_i \leq a_i$, we conclude that

$$E(D_{i+1}) \geq (1-a_i)\frac{\nu}{1-\nu}\,.$$

(b) Rearranging the terms in Equation (25), we have

$$\begin{aligned} D_{i+1} \quad \leq \quad & \max\left(0, \left(1 - \frac{1-B_i}{n}\right) D_i + \frac{B_i}{n} + \frac{1-B_i}{n} I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} x_i \right. \\ & \left. + \frac{1-B_i}{n} I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i \right). \end{aligned}$$

Squaring the latter equation on both sides and using the fact that B_i, $I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}}$ and $I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}}$ are binary-valued yields

$$\begin{aligned} D_{i+1}^2 \leq & \left(1 - \frac{1-B_i}{n}\left(2 - \frac{1}{n}\right)\right) D_i^2 + 2\frac{B_i}{n} D_i + \left(\frac{1-B_i}{n}\left(2 - \frac{1}{n}\right)\right) I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i \\ & + 2\frac{1-B_i}{n}\left(1 - \frac{1}{n}\right) I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} x_i D_i + \frac{1-B_i}{n^2} I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} {x_i}^2 + \frac{B_i}{n^2}\,. \end{aligned}$$

Taking expectation of the above equation, by Lemma 18, we have

$$\begin{aligned} E(D_{i+1}^2) \quad \leq \quad & \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)(1-\alpha)\right) E(D_i^2) \\ & + 2\left(\frac{\nu}{n} + \alpha\frac{1-\nu}{n}\left(1 - \frac{1}{n}\right)\right) E(D_i) + \frac{\nu + (1-\nu)E({x_i}^2)}{n^2}\,. \end{aligned}$$

We are now in an equivalent situation as in the proof of Theorem 7, right after Equation (22). Similary, we insert the result of (a) into the above equation, obtaining

$$\begin{aligned} E(D_{i+1}^2) \leq & \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)(1-\alpha)\right) E(D_i^2) \\ & + 2\left(\frac{\nu}{n} + \alpha\frac{1-\nu}{n}\left(1 - \frac{1}{n}\right)\right)(1-a_i)\frac{\nu + \alpha(1-\nu)}{(1-\nu)(1-\alpha)} + \frac{\nu + (1-\nu)E({x_i}^2)}{n^2} \\ \leq & \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)(1-\alpha)\right) E(D_i^2) + 2(1-a_i)\frac{(\nu + \alpha(1-\nu))^2}{n(1-\nu)(1-\alpha)} \\ & + \frac{\nu + (1-\nu)E({x_i}^2)}{n^2}\,. \end{aligned}$$

By the formula of the geometric series we obtain

$$\begin{aligned} E(D_i^2) \quad \leq \quad & \left(2(1-a_i)\frac{(\nu + \alpha(1-\nu))^2}{n(1-\nu)(1-\alpha)} + \frac{\nu + (1-\nu)E({x_i}^2)}{n^2}\right) \frac{1-b_i}{\frac{1-\nu}{n}(2 - \frac{1}{n})(1-\alpha)} \\ \leq \quad & \frac{(1-a_i)(1-b_i)(\nu + \alpha(1-\nu))^2}{(1 - \frac{1}{2n})(1-\nu)^2(1-\alpha)^2} + \frac{(1-b_i)(\nu + (1-\nu)E({x_i}^2))}{(2n-1)(1-\nu)(1-\alpha)}\,, \end{aligned} \tag{28}$$

where $b_i = \left(1 - \frac{1-\nu}{n}(2-\frac{1}{n})(1-\alpha)\right)^i$. We finally conclude

$$\begin{aligned}
\mathrm{Var}(D_i) &= E(D_i^2) - (E(D_i))^2 \\
&\overset{(27),(28)}{\leq} \frac{(1-a_i)(1-b_i)(\nu+\alpha(1-\nu))^2}{(1-\frac{1}{2n})(1-\nu)^2(1-\alpha)^2} + \frac{(1-b_i)(\nu+(1-\nu)E({x_i}^2))}{(2n-1)(1-\nu)(1-\alpha)} - (1-a_i)^2\frac{\nu^2}{(1-\nu)^2} \\
&\overset{(1)}{\leq} \gamma_i \frac{\nu^2}{(1-\alpha)^2(1-\nu)^2} + \rho(\alpha) + \delta_n\,,
\end{aligned}$$

defining $\gamma_i = a_i - b_i$, $\rho(\alpha) = \alpha\frac{(1-a_i)(1-b_i)(2\nu(1-\alpha)+\alpha)}{(1-\frac{1}{2n})(1-\nu)^2(1-\alpha)^2}$, and $\delta_n = \frac{(1-b_i)(\nu+(1-\nu)E({x_i}^2))}{(2n-1)(1-\nu)(1-\alpha)}$, where (1) can be verified employing some algebra and using the auxiliary formula Equation (24), which holds for all $0 < b_i < a_i < 1$. This completes the proof of (b).

Statements (c) and (d) are easily derived from (a) and (b) by noting hat $0 \leq a_i < 1$, $a_i \to 1$ for $i \to \infty$ and $\delta(n) \to 0$ for $n \to \infty$. This completes the proof of the theorem. ■

References

D. Angluin and P. Laird. Learning from noisy examples. *Machine Learning*, 2(4):434–470, 1988.

P. Auer. Learning nested differences in the presence of malicious noise. *Theoretical Computer Science*, 185(1):159–175, 1997.

M. Bailey, J. Oberheide, J. Andersen, Z. M. Mao, F. Jahanian, and J. Nazario. Automated classification and analysis of internet malware. In *Recent Adances in Intrusion Detection (RAID)*, pages 178–197, 2007.

M. Barreno, B. Nelson, R. Sears, A. Joseph, and J. Tygar. Can machine learning be secure? In *ACM Symposium on Information, Computer and Communication Security*, pages 16–25, 2006.

M. Barreno, P. L. Bartlett, F. J. Chi, A. D. Joseph, B. Nelson, B. I. Rubinstein, U. Saini, and J. D. Tygar. Open problems in the security of learning. In *AISec '08: Proceedings of the 1st ACM workshop on Workshop on AISec*, pages 19–26, New York, NY, USA, 2008. ACM.

M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010.

B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, 1992.

M. L. Braun, J. Buhmann, and K.-R. Müller. On relevant dimensions in kernel feature spaces. *Journal of Machine Learning Research*, 9:1875–1908, Aug 2008.

N. H. Bschouty, N. Eiron, and E. Kushilevitz. PAC learning with nasty noise. In *Algorithmic Learning Theory (ALT 1999)*, pages 206–218, 1999.

G. Cretu, A. Stavrou, M. Locasto, S. Stolfo, and A. Keromytis. Casting out demons: Sanitizing training data for anomaly sensors. In *Proc. of IEEE Symposium on Security and Privacy*, pages 81–95, 2008.

G. F. Cretu-Ciocarlie, A. Stavrou, M. E. Locasto, and S. J. Stolfo. Adaptive anomaly detection via self-calibration and dynamic updating. In *Recent Adances in Intrusion Detection (RAID)*, pages 41–60, 2009.

N. Dalvi, P. Domingos, M. Sumit, and S. D. Verma. Adversarial classification. In *In KDD*, pages 99–108. ACM Press, 2004.

O. Dekel and O. Shamir. Learning to classify with missing and corrupted features. In *International Conference on Machine Learning (ICML)*, pages 216–223, 2008.

O. Dekel, O. Shamir, and L. Xiao. Learning to classify with missing and corrupted features. *Machine Learning*, 2009.

P. Fogla and W. Lee. Evading network anomaly detection systems: formal reasoning and practical techniques. In *ACM Conference on Computer and Communications Security*, pages 59–68, 2006.

P. Fogla, M. Sharif, R. Perdisci, O. Kolesnikov, and W. Lee. Polymorphic blending attacks. In *Proc. of USENIX Security Symposium*, pages 241–256, 2006.

S. Forrest, S. Hofmeyr, A. Somayaji, and T. Longstaff. A sense of self for unix processes. In *Proc. of IEEE Symposium on Security and Privacy*, pages 120–128, Oakland, CA, USA, 1996.

A. Globerson and S. Roweis. Nightmare at test time: Robust learning by feature deletion. In *International Conference on Machine Learning (ICML)*, pages 353–360, 2006.

S. Hofmeyr, S. Forrest, and A. Somayaji. Intrusion detection using sequences of system calls. *Journal of Computer Security*, 6(3):151–180, 1998.

M. Kearns and M. Li. Learning in the presence of malicious errors. *SIAM Journal on Computing*, 22(4):807–837, 1993.

P. Laskov and M. Kloft. A framework for quantitative security analysis of machine learning. In D. Balfanz and J. Staddon, editors, *AISec*, pages 1–4. ACM, 2009. ISBN 978-1-60558-781-3.

P. Laskov, C. Schäfer, and I. Kotenko. Intrusion detection in unlabeled data with quarter-sphere support vector machines. In *Detection of Intrusions and Malware, and Vulnerability Assessment, Proc. of DIMVA Conference*, pages 71–82, 2004a.

P. Laskov, C. Schäfer, I. Kotenko, and K.-R. Müller. Intrusion detection in unlabeled data with quarter-sphere support vector machines (extended version). *Praxis der Informationsverarbeitung und Kommunikation*, 27:228–236, 2004b.

P. Laskov, C. Gehl, S. Krüger, and K. R. Müller. Incremental support vector learning: Analysis, implementation and applications. *Journal of Machine Learning Research*, 7:1909–1936, Sept. 2006.

A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava. A comparative study of anomaly detection schemes in network intrusion detection. In *Proc. of SIAM International Conference on Data Mining (SDM)*, 2003.

C. Leslie, E. Eskin, and W. Noble. The spectrum kernel: A string kernel for SVM protein classification. In *Proc. Pacific Symp. Biocomputing*, pages 564–575, 2002.

Z. Li, M. Sandhi, Y. Chen, M.-Y. Kao, and B. Chavez. Hamsa: fast signature generation for zero-day polymorphic worms with provable attack resilience. In *IEEE Security and Privacy*, pages 32–47, 2006.

N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear threshold algorithm. *Machine Learning*, 2:285–318, 1988.

D. Lowd and C. Meek. Good word attacks on statistical spam filters. In *Proceedings of the* 11^{th} *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 641–647, 2005a.

D. Lowd and C. Meek. Adversarial learning. In *Conference on Email and Anti-Spam*, 2005b.

M. Markou and S. Singh. Novelty detection: a review – part 1: statistical approaches. *Signal Processing*, 83:2481–2497, 2003a.

M. Markou and S. Singh. Novelty detection: a review – part 2: neural network based approaches. *Signal Processing*, 83:2499–2521, 2003b.

L. Martein and S. Schaible. On solving a linear program with one quadratic constraint. *Decisions in Economics and Finance*, 10:75–90, 2005.

Microsoft. Microsoft security intelligence report: January to June 2008. Microsoft Corporation, 2008.

M. Mohri and A. Rostamizadeh. Stability bounds for stationary φ-mixing and β-mixing processes. *J. Mach. Learn. Res.*, 11:789–814, March 2010. ISSN 1532-4435. URL `http://portal.acm.org/citation.cfm?id=1756006.1756032`.

D. Moore, C. Shannon, and J. Brown. Code-Red: a case study on the spread and victims of an internet worm. In *Proc. of Internet Measurement Workshop (IMW)*, pages 273–284, 2002.

K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Neural Networks*, 12(2):181–201, May 2001.

A. Nairac, T. N., R. Carr, S. King, P. Cowley, and L. Tarassenko. A system for the analysis fo jet vibration data. *Integrated Computer-Aided Engineering*, 1999.

B. Nelson and A. D. Joseph. Bounding an attack's complexity for a simple learning model. In *Proc. of the First Workshop on Tackling Computer Systems Problems with Machine Learning Techniques (SysML)*, Saint-Malo, France, 2006.

B. Nelson, M. Barreno, F. Chi, A. Joseph, B. Rubinstein, U. Saini, C. Sutton, J. Tygar, and K. Xia. Exploiting machine learning to subvert your spam filter. In *Proceedings of the First USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET'08)*, 2008.

J. Newsome, B. Karp, and D. Song. Paragraph: Thwarting signature learning by training maliciously. In *Recent Adances in Intrusion Detection (RAID)*, pages 81–105, 2006.

E. Parzen. On estimation of probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

R. Perdisci, D. Dagon, W. Lee, P. Fogla, and M. Sharif. Misleading worm signature generators using deliberate noise injection. In *Proc. of IEEE Symposium on Security and Privacy*, pages 17–31, 2006.

W. Polonik. Measuring mass concentration and estimating density contour clusters – an excess mass approach. *Annals of Statistics*, 23:855–881, 1995.

S. Rajasegarar, C. Leckie, M. Palaniswami, and J. Bezdek. Quarter sphere based distributed anomaly detection in wireless sensor networks. In *IEEE International Conference on Communications (ICC)*, pages 3864–3869, 2007.

K. Rieck and P. Laskov. Detecting unknown network attacks using language models. In *Detection of Intrusions and Malware, and Vulnerability Assessment, Proc. of 3rd DIMVA Conference*, LNCS, pages 74–90, July 2006.

K. Rieck and P. Laskov. Language models for detection of unknown attacks in network traffic. *Journal in Computer Virology*, 2(4):243–256, 2007.

K. Rieck and P. Laskov. Linear-time computation of similarity measures for sequential data. *Journal of Machine Learning Research*, 9(Jan):23–48, 2008.

K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov. Learning and classification of malware behavior. In *Detection of Intrusions and Malware, and Vulnerability Assessment, Proc. of 5th DIMVA Conference*, LNCS, pages 108–125, 2008.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

C. Shannon and D. Moore. The spread of the Witty worm. *IEEE Security and Privacy*, 2(4):46–50, 2004.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

Y. Song, M. Locasto, A. Stavrou, A. Keromytis, and S. Stolfo. On the infeasibility of modeling polymorphic shellcode. In *Conference on Computer and Communications Security (CCS)*, pages 541–551, 2007.

I. Steinwart, D. Hush, and C. Scovel. A classification framework for anomaly detection. *Journal of Machine Learning Research*, 6:211–232, 2005.

I. Steinwart, D. Hush, and C. Scovel. Learning from dependent observations. *J. Multivar. Anal.*, 100:175–194, January 2009. ISSN 0047-259X.

M. Sugiyama, M. Krauledat, and K.-R. Müller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8:1027–1061, 2007.

Symantec. Symantex report on the underground economy: July 07 to June 08. Symantec Corporation, 2008.

D. Tax and R. Duin. Data domain description by support vectors. In M. Verleysen, editor, *Proc. ESANN*, pages 251–256, Brussels, 1999a. D. Facto Press.

D. Tax and R. Duin. Support vector domain description. *Pattern Recognition Letters*, 20(11–13): 1191–1199, 1999b.

A. Tsybakov. On nonparametric estimation of density level sets. *Annals of Statistics*, 25:948–969, 1997.

C. van de Panne. Programming with a quadratic constraint. *Management Science*, 12:798–815, 1966.

V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

R. Vert and J.-P. Vert. Consistency and convergence rates of one-class svms and related algorithms. *J. Mach. Learn. Res.*, 7:789–814, December 2006. ISSN 1532-4435.

K. Wang and S. Stolfo. Anomalous payload-based network intrusion detection. In *Recent Adances in Intrusion Detection (RAID)*, pages 203–222, 2004.

K. Wang, G. Cretu, and S. Stolfo. Anomalous payload-based worm detection and signature generation. In *Recent Adances in Intrusion Detection (RAID)*, 2005.

K. Wang, J. Parekh, and S. Stolfo. Anagram: A content anomaly detector resistant to mimicry attack. In *Recent Adances in Intrusion Detection (RAID)*, pages 226–248, 2006.

C. Warrender, S. Forrest, and B. Pearlmutter. Detecting intrusions using system calls: alternative data methods. In *Proc. of IEEE Symposium on Security and Privacy*, pages 133–145, 1999.

D.-Y. Yeung and C. Chow. Parzen-window network intrusion detectors. In *Sixteenth International Conference on Pattern Recognition (ICPR)*, pages 385–388, 2002.