Security in Machine Learning: Measuring the relative sensitivity of classifiers to adversary-selected training data

Timothy Woods, Michael Evans, Dawn Rust, Ben Podoll
University of Minnesota
200 Union St SE
Minneapolis, MN 55455 USA
{wood0118, evan0341, rust0136, podo0029}@umn.edu

Abstract

In the spam domain, the data mining classifier Naïve Bayes has become the standard for use in commercial anti-spam products. It has therefore received much study from the research community, recently in terms of its susceptibility to adversarial attack and its poor defense against these attacks. Yet, little work has been done to show any sort of comparison of this popular classifier against other classifiers. Determined to find a better and more resilient classifier, we compared Naïve Bayes against three other classifiers. Our ensemble learning algorithm, AdaBoost, significantly outperformed the other classifiers in 29 out of 30 experiments. This could potentially drive further work into using ensemble learners for security in machine learning.

1 Introduction

1.1 Overview

We are interested in the intersection of security and machine learning. Specifically, how can an adversary affect a machine-learning system? Current research in this area works extensively with spam filters, with an adversary who supplies malicious emails to the system. These systems 'learn' from these new emails, which then affects their spam labeling going forward.

We compare the impact of attacks on classifiers that are used in these types of systems to identify relative vulnerability or robustness of these classifiers to these attacks.

1.2 The Importance of Security in Machine Learning

Many computer systems are developed for use in one context, and then are later used in a different one, resulting in some mismatches between the assumptions of the developer and the reality of the environment. When the transition is from a secure, controlled environment to a potentially hostile environment, this mismatch can result in significant security failures.

In the case of machine learning, most classification algorithms were developed to learn about passive target classes. When trained on data that contains some data that was crafted by an adversary, these algorithms are vulnerable to several types of attacks on integrity and availability[1]. In fact, controlling as little as 1% of the training data is sufficient in some cases[2].

It may seem odd that someone would use data provided by an adversary to train a system. However, for some real-world systems, such as systems that detect email spam, link spam, money laundering, credit card fraud, check fraud, insurance fraud, and other unwanted activities, the past actions of adversaries are a primary source of training data, and most systems are refreshed regularly so that they are always trained on the most recent examples of unwanted behavior. This provides sufficient influence for an adversary to attempt some attacks.

2 Background and related work

2.1 Security in Machine Learning

A plurality of the directly related work we have found has come from Joseph et al at the SecML research group (http://radlab.cs.berkeley.edu/wiki/SecML), part of the Reliable Adaptive Distributed Systems Laboratory (RAD Lab) at the University of California at Berkeley. Joseph et al has directly addressed the existence of the vulnerability of adaptive systems in security-sensitive environments [1, 3], developed a threat model for adaptive spam filters[3], and quantified the extent of data that an adversary would need to control in order to execute an attack [2], and identified what they consider to be the major open problems in the security of adaptive systems [4]. In their paper on open problems in this area, they name quantifying the influence of an adversary and establishing bounds on the cost and impact of an attack for categories of learners as key problems that should be addressed by researchers in this area. Although a formal proof of bounds or taxonomy of classifiers with regard to security is beyond the scope of this project, we believe that providing some experimental results will help advance the work toward developing them.

Several other researchers have developed experimental results relevant to this question, generally in the form of conducting attacks on specific systems. Most of the experimental work seems to have been conducted in the spam domain [2, 5, 6] , although at least one researcher reported results related to network traffic anomaly detection [7]. Joseph et al notes that there is currently no (accepted) theoretical analysis or interpretation of these attacks

Although Joseph et al seeks to identify the impact of adversarial action on learning systems, and possibly to prevent it, other researchers have approached the problem of learning about potentially deceptive adversaries by incorporating explicit models of their adversaries into their learning algorithms, including explicit models of their adversaries' learning capabilities [8, 9].

This impasse is the definition of a Nash equilibrium in non-cooperative game theory [10]. Many researchers dealing with adversarial (especially zero-sum) domains have sought to calculate equilibrium strategy sets directly and employ the constituent strategies. This approach is by its nature conservative, because a system employing it may pass up opportunities to exploit non-optimal opponents. It is also often prohibitively expensive to calculate the equilibria of a non-trivial game.

2.2 The Spam Domain

Classification algorithms have been widely used for spam filtering. There is considerable work on developing ways to separate good email (ham) from unsolicited junk message (spam) by using classifiers to identify distinguishing features of the contents of messages. This is the area where we will test our attacks.

The spam-filtering domain also has considerable work on non-adaptive filtering, and on domain-specific challenges such as recognizing messages embedded in image files. The work to extract meaningful features from obfuscated content (such as dealing with the sextillion+ ways to spell "Viagra" so that a human can read it -- http://cockeyed.com/lessons/viagra/viagra.html), and any approaches to filtering that are based on non-content features (such as analyzing the SMTP path or analyzing the user's social network) are out of the scope of this project.

Threat model

The threat model for adaptive spam filters developed by Barreno [3] classifies attacks according to whether they are Causative or Exploratory, Targeted or Indiscriminate, and whether they are aimed at disrupting Integrity or Availability. A Causative attack aims to cause misclassification of a message or set of messages, whereas an Exploratory attack aims to determine the classification of a message or set of messages. Attacks on Integrity seek to affect or reveal the classification of spam, while attacks on Availability seek to affect or reveal the classification of ham.

Adversary capabilities

The most basic ability of a spammer is to send spam that is caught by the filter (or user) and labeled as spam. Other capabilities are possible (see section 5.0, Future Work), but all of the attacks here rely exclusively on a capability of having arbitrary messages classified as spam.

3 Experimental approach

3.1 Dataset

The dataset we are using is a set of email messages from the Enron Corporation, with each message labeled either "spam" or "ham", that was prepared for the Spam Track of the 2005 Text REtrieval Conference (TREC)[11]. Each email is in its own text file, in its original form. We will first need to preprocess these emails into a matrix format that a machine-learning algorithm can understand.

We treat each email as a bag of words extracted from the content of the message. Network information (such as originating IP addresses and other message metadata) is excluded from the feature set.

Preprocessing

We used a common data-mining tool called Weka[12] to train and test the classifiers, as it provides a common interface for all of the target classifiers. The dataset needed to go through a series of steps to be ready for processing in Weka. The original emails were converted to a matrix of size n by m, where each row represents an email and each column represents a word found in any email. In size, n will be the number of emails we are working with, and m is the number of distinct words found in all the emails. The value in each element of the matrix is the number of times that the word indicated by that column appeared in the message indicated by that row.

Problems/Limitations

Due to the large size of dataset, some filtering needed to take place to allow for reasonable processing times. We first took a random subset of our dataset, keeping the proportion of spam to ham emails the same as the full set of emails. We removed any files we believed to have attachments by filtering emails of a predetermined size. That removed 'words' that were actually long streams of data in character format. That helped significantly, but we still had too many columns; hundreds of thousands. We created a function in Matlab to identify columns (words) that were insignificant. The function counted how many rows (emails) had any value of that particular word, and if the total fell under a certain threshold, we removed it. This improved our column count significantly. This resulted in a small yet representative dataset that we converted into a Weka format.

3.2 Attacks

Attack selection

In considering the type of attacks we would test, we needed to consider a number of factors. First, we needed to know which ones would be tractable within our available processing time based on both the time required to generate attack data and the time required to perform the additional train-test cycles. Second, we want attacks that are plausible in our test domain, both because they require knowledge that the attacker might reasonably be expected to know, and because their cost and value to the attacker would make them attractive. The dictionary and mimicry attacks require only the knowledge of the general words in use, and a probabilistic knowledge of a single message (respectively) so these are quite reasonable in terms of attacker knowledge. The cost of carrying out the attacks is just the sending of some emails that are labeled as spam, which is very inexpensive to the attacker, and the value is either the deactivation of the spam filter (for dictionary attacks) or the loss of a valuable email message (for the mimicry attacks). We also expect that these attacks are different enough that the various algorithms will exhibit different relative degradation rates as the percentage of poisoned training data increases. That is, we expect that the vulnerability to these attacks is not highly correlated among the algorithms.

Mimicry attack

In this attack, the adversary has a target good (ham) email it is trying to have misclassified as spam. This is a type of allergy attack in which the target message is mimicked as closely as possible. All or a portion of the target email's content is known, allowing the adversary to send copies of that email, again, from a blacklisted IP address, forcing the email to be classified as spam.

To create this attack, we take a ham email that is not included in the dataset we are using, and create a number of attack emails. When creating an attack email, each word from the target email is included with probability p. If it is not included, a random word from the Linux dictionary was included.

3.3 Target Classifiers

We tested some basic classifiers (Naïve Bayes, decision trees [C4.5], boosted decision stumps, and Ripper) as spam filters. Tests were run both on the existing data (a baseline measure) and with an active adversary. These algorithms were chosen because all of them have been previously used or proposed for use as spam detection algorithms[13], and also because they represent a sampling of different families of classifiers (see below) that have different levels of expressiveness and robustness to noise. In addition, all of these algorithms are commonly studied and easily understood by a person who is new to machine learning. We hope this will make our experiments accessible to many readers.

Naïve Bayes

Naïve Bayes is widely used for spam detection and other applications due to its speed, ease of implementation, and generally good results. An example of a simple Naïve Bayes spam filter that is in wide use is SpamBayes. Although simple, Seewald found the basic learner in SpamBayes to perform better than some more complex, specially tuned variations of Naïve Bayes [14]. The systems judged inferior to SpamBayes by Seewald include SpamAssassin, the spam filter is used by the University of Minnesota Computer Science Department.

We were also interested in Naïve Bayes because Nelson performed an analysis of the vulnerability of SpamBayes to mimicry attacks [2], and we wanted to include a similar classifier as a way of reproducing his results in a situation that allowed comparison with other classifiers.

AdaBoost with Decision Stumps

We wanted to include at least one classifier based on ensemble learning, and to stick to very basic classifiers. AdaBoost using Decision Stumps as the weak learners is a very simple, perhaps simplistic, representative of the class of ensemble learners. More sophisticated boosting algorithms and ensemble classifiers exist, but since we were hoping to capture a comparison between very basic classifiers, we felt this was a good choice.

C4.5 Decision Tree

Decision Trees are another type of basic classifier that is widely used. We chose the C4.5 algorithm for building the decision trees, rather than a simpler decision tree building algorithm such as ID3, because C4.5 deals better with real-valued attributes yet is still easy enough to understand. C4.5 also includes some regularization (by pruning of the resulting decision tree), and Barreno has suggested that various regularizing mechanisms would be possible defenses against some attacks [3].

Ripper

Ripper is a rule-generating algorithm that is designed to be especially well-suited for classification problems with a rare class. Our test data set has roughly balanced classes so this strength may not be evident, but we wanted a representative of algorithms that work well with rare classes in our set of classifiers. Rule-generating systems are related to decision trees, but the methods used by Ripper and C4.5 to produce the decision points are quite different.

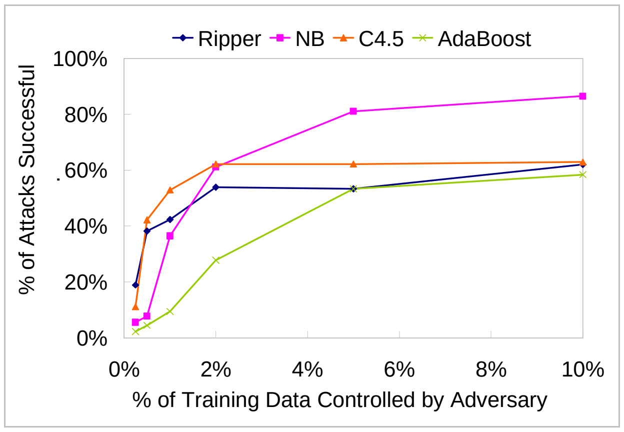

Figure 1: Attack success against each classifier by the percentage of training data controlled by the adversary, aggregation of all tests performed

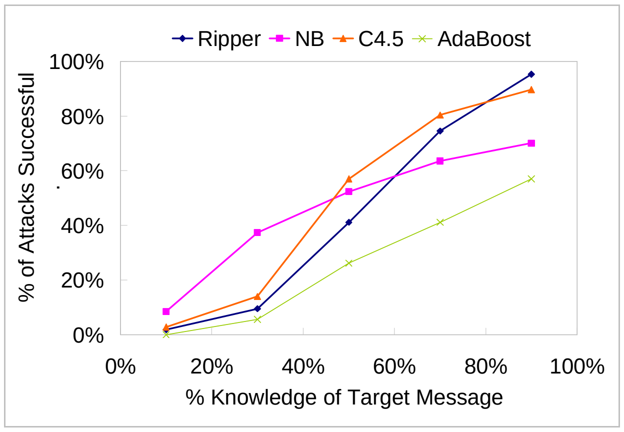

Figure 2: Attack success against each classifier by the adversary's percentage chance of guessing each token in the target messages

4 Experimental Results

We first review the overall success of the attacks and the net performance of each algorithm on our chosen attacks, and then later, we will see how the classifiers performed differently based on the variables being tested, such as percent known of the target email, or percent of data controlled by the adversary. This provides a useful look at the potential strengths and weaknesses of each classifier.

4.1 Baseline classification accuracy

Before adding any poisoned data, we ran the classifiers on the clean data set to see how effective each one was at the underlying classification task, using ten-fold cross validation to evaluate the effectiveness of each classifier. Although the later attack experiments do not look at the impact on overall classification accuracy (only at the success or failure of the attack on the target message), the base rate of success is important because this would be one of the key measures for evaluating the fitness of a classifier for any particular task.

4.2 Overall Performance

We were able to successfully recreate the findings of the Joseph et al research group [1, 3], Our mimicry attack was able to fool Naïve Bayes with a very small portion of the data controlled, similar to their findings. This is important; we were not sure how their results would transfer to a simpler implementation of a Naïve Bayes algorithm. They ran their experiments on a commercial product that used NB, while we were running a data mining tool using NB. In addition, the other classification algorithms we ran had similar results. Our results will be useful to the community, as they clearly fit within the norms of other experiments while adding value through testing these other classifiers.

Classifier	***% of Attacks Successful***	***Baseline Accuracy***
AdaBoost	26.0%	96.51%
C4.5	48.9%	96.32%
Naïve Bayes	46.4%	94.70%
Ripper	44.7%	96.82%
Grand Total	41.5%	96.09%

It is clear to see that while Naïve Bayes is not the worst algorithm for resilience against adversary attack, it likely is not the best. The AdaBoost classifier performed significantly better than the rest, implying that it may be worth doing a more thorough analysis on ensemble for use in this domain.

We ran 30 different experiments with various combinations of our two variables, percent of target email known and percent of data controlled. Each experiment was run on 17 different target emails. AdaBoost had the best average classification performance in 29 of the 30 experiments, usually with a significant margin. This is substantial, not dominance on a few experiments, or with select parameters, but better in almost every experiment. These results point to ensemble learning algorithms as possible good choices when considering security in machine learning algorithms.

4.3 Sensitivity to adversary control

C4.5 and Ripper

These two algorithms showed similar vulnerability to similar combinations of attack parameters. This is not very surprising since both algorithms are similarly expressive in terms of the decision boundaries they create. Both algorithms create rectilinear partitions of the attribute space, and both perform some explicit feature prioritization as part of building these boundaries.

These two classifiers had relatively greater vulnerability at low levels of adversary control (less than 1%, see **Error! Reference source not found.**), but then seemed to plateau after about 2% of control while AdaBoost and Naïve Bayes continued to degrade. The likely reason for this is that these classifiers rely on only a subset of features for each decision. This means that a large amount of influence may be achieved with little data if a particular feature can be subverted sufficiently to create a decision node/rule that uses it. However, some messages can be classified based on factors such as the words not included (eg, “All emails without the word Viagra are not spam”). Mimicry attacks would need to flip the significance of this feature for the entire dataset, and not just for a localized section of the boundary, if they wanted to affect these messages. Of course, this is possible with sufficient data, but it is likely that our upper bound of 10% adversary control was not high enough for this kind of impact.

Naïve Bayes and AdaBoost

Naïve Bayes and AdaBoost displayed continuous degradation over the range of adversary control that we tested. For Naïve Bayes this probably reflects its use of all features for each decision as well as the gradual shift in the overall ratio of spam to ham. AdaBoost degrades continuously, but at a much slower rate. The continuous degradation probably reflects the fact that AdaBoost’s ensemble of classifiers is likely to use a large number of attributes for each decision, so it is affected by each attack message. We do not have a theory as to the overall slower rate of degradation, but consider it an extremely interesting area for future research.

4.4 Sensitivity to adversary knowledge

C4.5 and Ripper

Despite their apparent resilience to increasing adversary control of the training data, C4.5 and Ripper were relatively more sensitive to the adversary’s level of knowledge of the target message. They performed relatively well when the adversary had little (30% or less) knowledge of the contents of the target message, but degraded quickly after this point and had almost no success when the adversary had a very high degree of knowledge (70% to 90%) of the contents of the message. This is likely because targeting the relevant attributes made it easier to shift small, local sections of the decision boundary by creating small but high-purity leaf nodes (rather than having to shift or “pollute” the purity of larger intermediate nodes). Because the data is in a high-dimensional space it is very likely that the decision boundary will be “close” to any given point in at least one dimension, so very small shifts can be effective.

Naïve Bayes and AdaBoost

In contrast to C4.5 and Ripper, Naïve Bayes applies information about every attribute to every decision, so that its behavior is affected even when the most relevant features for the target message are not known. However, the sheer number of features in use seems to reduce the effectiveness of subverting the especially relevant ones. In effect, Naïve Bayes has no localized decision making because it assumes that all attributes are independent, while the rectilinear partitioning algorithms are very location-aware and more susceptible to local effects. AdaBoost with decision stumps is as expressive as decision trees and can capture local impacts, but it seems to be resilient to detailed adversary knowledge, especially when the percentage of adversary control is lower. This may be because the method of weighting the weak learners gives decision stumps that classify a broader range of points correctly a greater weight when voting, thus providing some regularization against localized overfitting.

4.5 Interaction between attack parameters

For all classifiers tested, the attacks succeeded 100% of the time when 10% of the training data was controlled and 90% of the contents of the target messages were known to the adversary, and 0% of the attacks succeeded when the attacker controlled only 0.25% of the training data and knew only 10% of the message contents. This provides us with an interesting range of the combinations of parameters that shows how the different classifiers are affected by different combinations of adversary control and knowledge.

The results for Naïve Bayes and AdaBoost seem to indicate a continuous tradeoff between the two attack parameters. In contrast, C4.5 and Ripper seemed to be more affected by adversary knowledge, while the percentage of adversary control seems to have sharp and perhaps discontinuous effects. Based on the results we see here and the nature of decision trees and rule-based algorithms, we consider it likely that if the percentage of adversary control were increased beyond 10% we would find additional sharp rises in the influence of the attackers followed by plateaus, rather than a smooth degradation.

5 Future Work

Additional attacks

There exists a large variety of other attacks on adaptive systems in general and spam filters in particular, and experiments evaluating the relative robustness of various classifiers to those attacks would be interesting. The attack evaluated in this paper was a causative attack on availability of the system. Other attacks include exploratory attacks to determine the state of the classifier, and attacks in the integrity of the system (i.e., on getting spam past the filter) such as desensitization attacks. In addition, other causative attacks on availability are possible. For example, the mimicry attack is not necessarily the most effective way to target a specific message, depending on the particular classifier.

Additional classifiers
The classifiers we chose are classic ones, heavily studied and in wide use. It is possible that other classification algorithms are more robust (or less robust) to adversarial influence, and we currently do not have any method for identifying them through static analysis of the algorithms.

It would also be instructive to compare classifiers from the same family, and to test identical classifiers with different parameters. This might provide additional insight into the question of *why* one particular classifier performs better or worse than another does.

In particular, we consider the success of AdaBoost in our experiments good reason to focus on ensemble classifiers and determine whether they have some inherent robustness against mimicry attacks and possibly other causative attacks as well.

Alternative adversary capabilities
In our experiments we assume that the adversary can add arbitrary spam-labeled messages to the training data. An attacker might have other capabilities, such as adding arbitrary ham-labeled messages to the training data. To understand this, it is important to realize that many anti-spam systems are trained using messages classified by all users of the email system, so a spammer could create a Gmail account (for example), and then send email messages to this account and mark it as “not spam”. There are obvious countermeasures that could be used to identify such accounts if they were used heavily, but a large number of low-volume accounts would be very difficult to detect, and would give spammers significant influence over the anti-spam system *even if those accounts never sent a single spam message*. The extent to which a spammer could inflict damage on a spam filter using this technique is unclear, and experiments to help quantify the risk would be valuable.

Other data sets and domains
Data from different domains exhibits different structures, so it is possible that the effect of poisoning data sets from different domains would produce different results. If datasets from different domains exhibit different levels of susceptibility to poisoning, it might provide some insight into how to identify vulnerable domains or how to condition data prior to training to make classifiers more robust against poisoning.

Data preprocessing defenses
For some existing systems, it would be very expensive to change the type of classifier in use, or the particular classifier might have been chosen for reasons that outweigh the risk of being attacked. In these cases, it might be desirable to find defenses that operate by transforming or conditioning the data. Some existing techniques for preprocessing data to improve classifier performance might provide some robustness against some types of attacks. Some examples worthy of investigation are dimensionality reduction techniques, which might weed out noise attributes and thus prevent them from being used as attack vectors, and domain-relevant feature transformations, such as inverse-document frequency for text data. These techniques effectively apply heuristics about what data is likely to be useful for learning, thus potentially limiting the effect of data designed to subvert the learning process. It is not clear whether these techniques would be effective, or if so whether they would provide the same benefits for all classifiers, but the potential benefit, especially to legacy systems, makes it a worthwhile area of investigation.

Developing more secure classifiers
Designing new classification algorithms that are less vulnerable to attack (to specific attacks or to whole classes of attacks) would be the ideal solution for securing new systems that use classification. Machine learning and classification in particular are already widely used in adversarial environments, and more secure classification algorithms would almost certainly have a benefit to many of these fields. Also, based on what we know about the benefits of developing secure software, it is plausible that algorithms that learn in a way that is robust against adversarial attacks would also have other desirable characteristics. In particular, it seems that more secure classifiers would need to be more robust against noisy data and disruption by outliers at a minimum, and these are generally desirable properties in many applications.

6 Conclusion

Security against manipulative attacks is simply not considered in the design of most systems that employ classification or other types of machine learning. However, we have shown that even when classifiers have similar performance on the basic classification task, they may have significantly different susceptibilities to a particular attack, and they may have different security considerations in terms of which adversary capabilities are the most likely to be damaging.

Given AdaBoost’s overall success and domination of each individual scenario, we consider investigation of the potential robustness of ensemble classifiers to attack to be a priority for future research in the security of machine learning.

It is the position of the authors that the vulnerability of a learning algorithm to manipulation by an adversary should be a primary consideration in designing or selecting an algorithm for use in any environment where an adversary with sufficient capabilities to execute an attack is known to exist. However, designing and developing secure adaptive systems is currently a difficult and uncertain task. In addition to the future research directions described above, an awareness of security issues relevant to machine learning should be integrated into basic classes and texts on classification, data mining, and other machine learning related topics.

Acknowledgements
The authors wish to thank Nick Hopper and Arindam Banerjee for their comments and ideas, which have helped us understand the problem and motivated our research.

% Control	% Known	Ripper	NB	C4.5	AdaBoost
0.0025	10%	0.0%	0.0%	0.0%	0.0%
	30%	5.6%	0.0%	5.6%	0.0%
	50%	5.6%	5.6%	5.6%	0.0%
	70%	11.1%	11.1%	5.6%	0.0%
	90%	72.2%	11.1%	38.9%	11.1%
0.005	10%	0.0%	0.0%	0.0%	0.0%
	30%	5.6%	5.6%	5.6%	0.0%
	50%	16.7%	11.1%	27.8%	5.6%
	70%	66.7%	11.1%	77.8%	5.6%
	90%	100.0%	11.1%	100.0%	11.1%
0.01	10%	0.0%	0.0%	0.0%	0.0%
	30%	11.8%	11.8%	11.8%	5.9%
	50%	23.5%	11.8%	52.9%	5.9%
	70%	76.5%	58.8%	100.0%	5.9%
	90%	100.0%	100.0%	100.0%	29.4%
0.02	10%	5.6%	11.1%	0.0%	0.0%
	30%	5.9%	11.1%	27.8%	5.6%
	50%	61.1%	83.3%	83.3%	11.1%
	70%	94.4%	100.0%	100.0%	33.3%
	90%	100.0%	100.0%	100.0%	88.9%
0.05	10%	5.6%	11.1%	11.1%	0.0%
	30%	5.6%	94.4%	16.7%	11.1%
	50%	55.6%	100.0%	83.3%	55.6%
	70%	100.0%	100.0%	100.0%	100.0%
	90%	100.0%	100.0%	100.0%	100.0%
0.1	10%	0.0%	29.4%	5.9%	0.0%
	30%	23.5%	100.0%	16.7%	11.1%
	50%	83.3%	100.0%	88.9%	77.8%
	70%	100.0%	100.0%	100.0%	100.0%
	90%	100.0%	100.0%	100.0%	100.0%

Figure 3: Full results of attacks for all classifiers and attack scenarios (% of attacks successful)

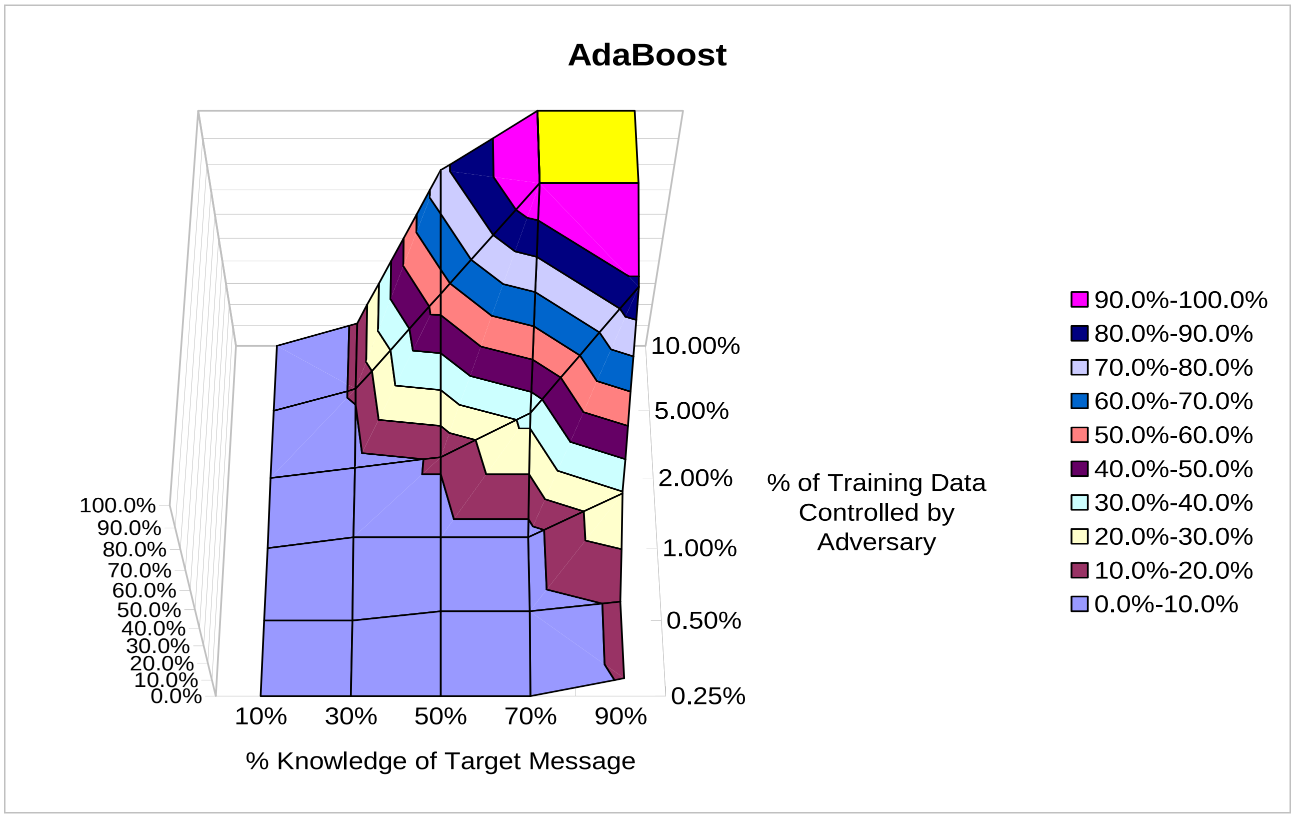

Figure 4: Surface plot of results for AdaBoost

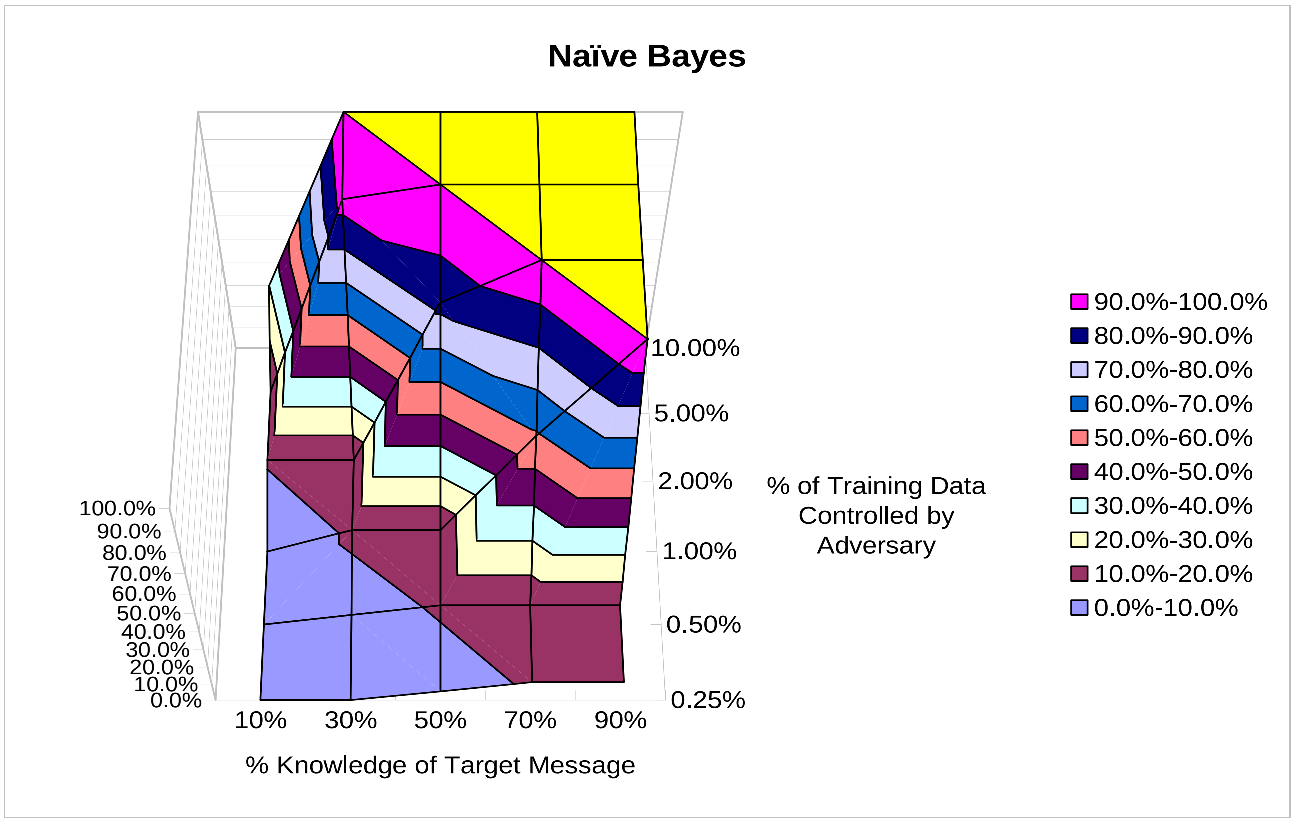

Figure 5: Surface plot of results for Naive Bayes

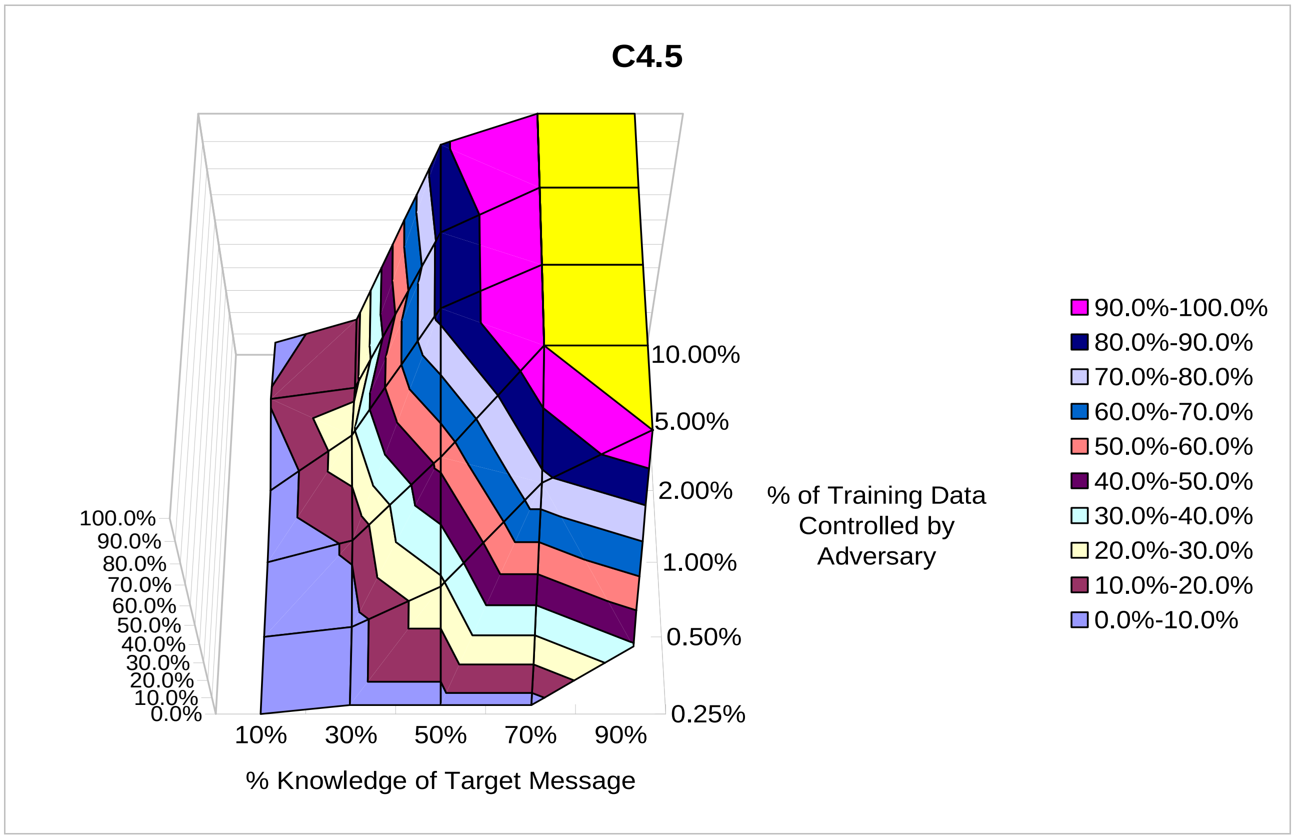

Figure 6: Surface plot of results for C4.5

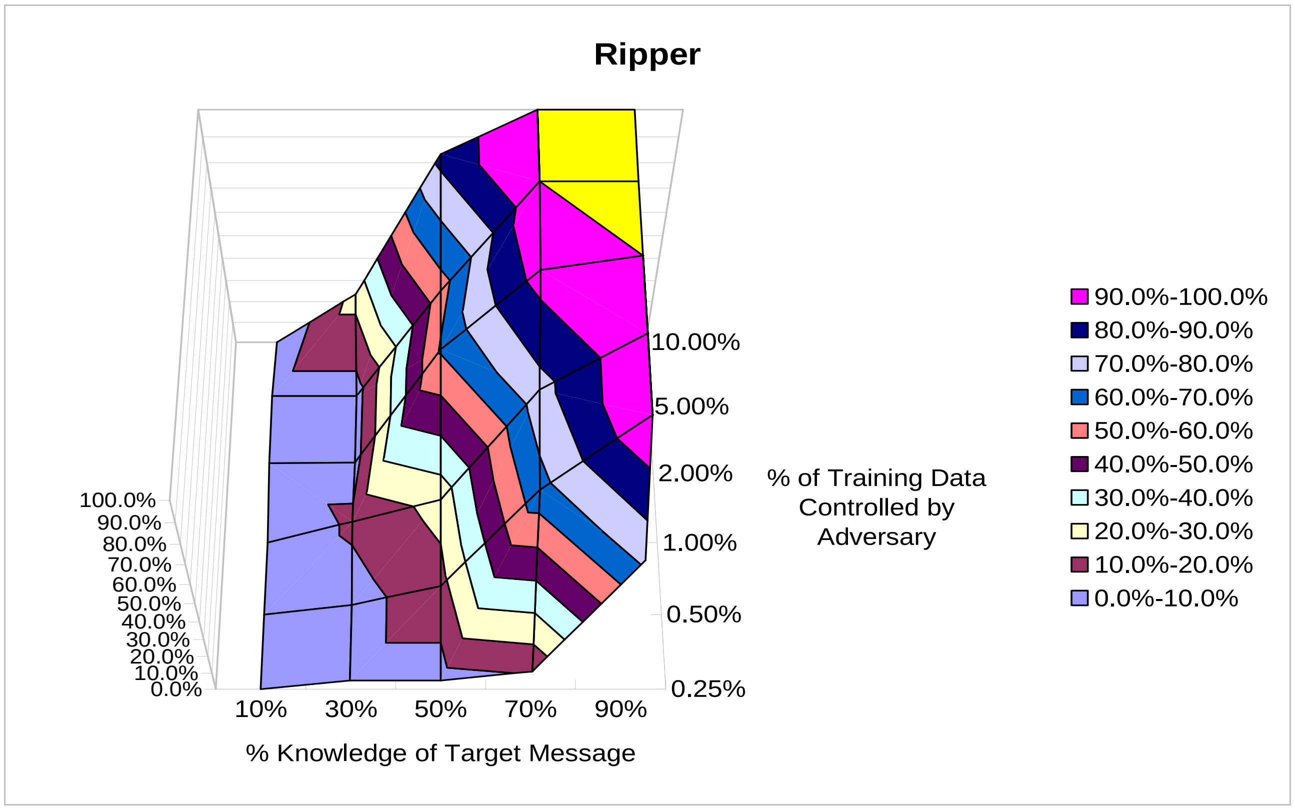

Figure 7: Surface plot of results for Ripper

References

[1] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. D. Tygar, "Can machine learning be secure?," in *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, Taipei, Taiwan, 2006, pp. 16-25.

[2] B. Nelson, M. Barreno, F. J. Chi, A. D. Joseph, B. I. P. Rubinstein, U. Saini, C. Sutton, J. D. Tygar, and K. Xia, "Exploiting machine learning to subvert your spam filter," in *Proceedings of the 1st USENIX Workshop on Large-Scale Exploits and Emergent Threats*, San Francisco, California, 2008, pp. 1-9.

[3] M. A. Barreno, "Evaluating the Security of Machine Learning Algorithms," EECS Department, University of California, Berkeley, 2008.

[4] M. Barreno, P. L. Bartlet, F. J. Chi, A. D. Joseph, B. Nelson, B. I. P. Rubinstein, U. Saini, and J. D. Tygar, "Open Problems in the Security of Learning," in *The First ACM Workshop on AISec, 2008* Alexandria, VA, 2008.

[5] D. Lowd and C. Meek, "Adversarial learning," in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, Chicago, Illinois, USA, 2005, pp. 641-647.

[6] G. L. Wittel and S. F. Wu, "On attacking statistical spam filters," in *Proceedings of the Conference on Email and Anti-Spam*, Mountain View 2004.

[7] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," *ACM SIGCOMM Computer Communication Review*, vol. 34, pp. 219-230, October 2004.

[8] J. M. Vidal and E. H. Durfee, "Learning Nested Models in an Information Economy," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 10, pp. 291-308, 1998.

[9] D. Egnor, "Iocaine Powder," *International Computer Games Association Journal*, vol. 23, pp. 33-35, 2000.

[10] J. Nash, "Non-Cooperative Games," *The Annals of Mathematics*, vol. 54, pp. 286-295, 1951.

[11] G. V. Cormack and T. R. Lynam, "Spam corpus creation for TREC," in *The Second Conference on Email and Anti-Spam*, Stanford University, Palo Alto, CA 2005.

[12] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, Second ed. San Francisco: Morgan Kaufmann, 2005.

[13] E. Blanzieri and A. Bryl, "A Survey of Learning-Based Techniques of Email Spam Filtering," Informatica e Telecomunicazioni, University of Trento, Technical Report DIT-06-056, 2006.

[14] A. K. Seewald, "An evaluation of Naive Bayes variants in content-based learning for spam filtering," *Intelligent Data Analysis*, vol. 11, pp. 497-524, 2007.