


Boldt, Martin; Jacobsson, Andreas, Lavesson; Niklas ; Davidsson, Paul.
"Automated Spyware Detection Using End User License Agreements"
*2nd International Conference on Information Security and Assurance, Busan, Korea 2008*

# Automated Spyware Detection Using End User License Agreements

Martin Boldt, Andreas Jacobsson, Niklas Lavesson, & Paul Davidsson
*Department of Systems and Software Engineering, School of Engineering,*
*Blekinge Institute of Technology, 372 25 Ronneby, Sweden*
*{martin.boldt;andreas.jacobsson;niklas.lavesson;paul.davidsson}@bth.se*

## Abstract

*The amount of spyware increases rapidly over the Internet and it is usually hard for the average user to know if a software application hosts spyware. This paper investigates the hypothesis that it is possible to detect from the End User License Agreement (EULA) whether its associated software hosts spyware or not. We generated a data set by collecting 100 applications with EULAs and classifying each EULA as either good or bad. An experiment was conducted, in which 15 popular default-configured mining algorithms were applied on the data set. The results show that 13 algorithms are significantly better than random guessing, thus we conclude that the hypothesis can be accepted. Moreover, 2 algorithms also perform significantly better than the current state-of-the-art EULA analysis method. Based on these results, we present a novel tool that can be used to prevent the installation of spyware.*

## 1. Introduction

The occurrence of spyware in user-oriented applications widely available over the Internet has become more and more commonplace during recent years. From a user perspective, spyware is inherently bad since it often degrades computer performance and stability, and violates the users' right to privacy. Due to that spyware typically is difficult to remove once it has entered a system and because current protection mechanisms can at best immunize spyware components, the need for new and more efficient spyware protection measures is accentuated. Ideally, countermeasures should not only remove unwanted software, but also prevent spyware from entering computers thus stopping any damage before it can actually occur. In this study we try to take advantage of the fact that, contrary to that of the creators of malware like viruses and worms, the vendors of spyware-hosting applications usually try to pose their software as legitimate. There are typically two objectives that counteract for these vendors in that they want users to download and install applications that covertly install spyware but without any legal consequences for the vendor. The most common solution seems to be to mention in the End User License Agreement (EULA) that spyware will indeed be installed but to give this information in a way most users find very hard to understand and time consuming to read. Not to mention that many EULAs contain thousands of words and even the EULAs of legitimate software could be hard to interpret for regular users due to the extensive use of legal terms. We therefore address the spyware problem by mining EULAs of both legitimate software and spyware-hosting applications looking for patterns in order to determine if it is possible to detect from the EULA whether the associated software hosts spyware or not. A data mining approach is in essence ideal for this problem since it can be used to sift through large amounts of data, in this case large text documents, to extract relevant information, e.g., combinations of words or phrases that are predominantly used in spyware EULAs. This paper is organized as follows. In the next section we present the problem domain and review the different existing definitions of spyware. Section 3 then reviews related work and the subsequent section describes the data gathering process, the experiment and the obtained results. In Section 5 we discuss some consequences of the results and present a novel tool for preventing spyware from being covertly installed and finally, we draw conclusions and give pointers to future work in Section 6.

## 2. Spyware

Spyware exists because information has value and is most often bundled as a hidden component of popular and free software widely available for download [13]. Some common examples of applications hosting spyware are file-sharing tools and instant messaging clients. Spyware is typically designed to collect user information usable for marketing campaigns, e.g., via toolbars, pop-up windows and spam E-mail messaging. Amazingly, close to 88% of consumer PCs was found to be infected by spyware and an average of 20 instances of spyware was found for every 1000th enterprise computer scanned in 2006 [34]. One report claims that the spyware industry earned over $2 billion distributing and installing software that monitored and reported on its victims in 2004 [35]. The anti-spyware industry, on the other hand, "only" earned an estimated $100 million the same year although its revenue is projected to reach $1.2 billion by 2010.

### 2.1 Definition

In general terms, spyware can be defined as a category of software that collects personal information about users without their informed consent [13]. Although, it has proven difficult to agree on a more formal and precise definition of spyware, it is clear that such software can be conceptually placed between ordinary (or legitimate) software and malicious software (malware) [24]. In contrast to spyware, the definition of malware and its subclasses (e.g., virus programs and worms) has proven more successful. One reason is that there is a distinct difference between legitimate software and its malicious (and illegitimate) counterpart [20]. Typically, such a difference cannot be found between ordinary software and spyware programs. The latter category is usually covertly bundled with seemingly "ordinary and legitimate" software and has a hidden functionality, which may be to, e.g., monitor Web surfing, display messages on the screen, and collect personal details about users and programs. The difficulty in making a clear distinction between legitimate and illegitimate software is manifested mainly within the anti-spyware industry. This has resulted in vast numbers of computer users experiencing loss of control over personal information, decreased computer performance and stability, and circumvented system integrity [15][16][23].

### 2.2 Countermeasures

Anti-virus techniques have traditionally been the most frequently used type of countermeasure against spyware, even though these techniques were originally designed for a radically different problem, i.e., separating illegitimate virus programs from legitimate software [22]. Due to the inability of anti-virus programs to properly define, discern and remove spyware, the threat of spyware is more considerable than ever. As a conclusion, instead of relying on techniques not originally designed for the versatility and ambiguity of the spyware problem we need new countermeasures that properly detect the particular characteristics of the commercially-motivated spyware. Adaware is perhaps the most popular specialized anti-spyware product [29]. It can be used to scan a system for spyware and returns the number of hits (occurrences of spyware) found, sorted into a number of categories, e.g., tracking cookies and modem hijackers. However, whereas Adaware and most anti-virus products are reactive, thus trying to remove something which has already infected a system, we examine the possibility of a preventive approach that detects the presence of spyware by examining the EULA, even before the application is installed.

### 2.3 The state-of-the-art

One way of detecting spyware could be to analyze its EULA. On this topic, there are a few applications available today. In fact, we have found one web based tool for analyzing the content of a EULA, namely the EULA analyzer [12]. The user pastes the content of a EULA into a text field on the EULA analyzer web page and presses submit in order to get the results..The analyzer sifts through the EULA text and assigns a credit point that indicates the likelihood of spyware inclusion in the particular application and also shows some statistics about the number of words, phrases, and the level of language complexity. However, it is important to notice that the credit point merely corresponds to the number of spyware related keywords or phrases that are found, i.e. a word count; there is no automatic learning of new patterns, there is no way to represent, e.g., non-trivial rules, and there is no software classification. A low credit point indicates legitimate software and high points indicates spyware. However, the credit limit that separates bad EULAs from good EULAs is dependent on the particular set of submitted EULAs and has to be manually found by the user, e.g., using a linear search, in order to be able to use the EULA analyzer for classifying software application based on their corresponding EULAs. Ultimately this means that the accuracy of this product in deciding whether application is good or bad depends on the user. Another product, that is very similar with regard to the EULA analysis technique, is EULAlyzer [36]. Unlike the web based EULA analyzer, EULAlyzer is an installable application. Unfortunately both state-of-the-art tools are proprietary and thus it is impossible to know for sure which techniques or algorithms are used however, the tool descriptions indicate that they rely on a static set of keywords.

## 3. Related Work

Within the research areas of information security and privacy, several studies on spyware have been presented. Sariou and Gribble introduce methods for measuring spyware activity in heavily trafficked networks [13]. The occurrence of spyware in popular file-sharing clients as well as the effects thereof has been outlined in a suit of papers. For instance, there are some initiatives exploring the purposes behind spyware as well as its commercial impact [17][18]. Fox investigates how user behavior is affected by the occurrence of spyware in home and work computers [19]. Robertson classifies five major classes of spyware and investigates how these classes affect computer usage [20]. Numerous white papers, technical reports, and surveys on various aspects of spyware are available [21][22][23]. McCardle presents an in depth analysis of spyware capabilities and discusses various technical means of protecting operating systems [22]. Townsend elaborates

on how spyware infected applications may violate corporate security policies and procedures [23]. Good et al. investigate the fact that users agree to accept spyware as part of a software bundle as a cost associated with gaining the functionality they desire and demonstrate that interface design can be a significant factor in eliciting informed consent to software installation [21]. Notably, very few initiatives on spyware countermeasures exist, although some that have been outlined include, e.g., anti-virus techniques [23] or user participation [24]. We have not been able to identify any studies which apply data mining algorithms on EULAs to find patterns that could be used to distinguish between good and bad software applications..Much work has been done in the related area of E-mail filtering, i.e., the classification of E-mail messages as legitimate or spam depending on the subject, body or other properties, using different mining algorithms, e.g.; rule learners, support vector machines, instance-based learners, decision trees, and stacking [5][6][7][8][9]. A more recent study investigates the performance of random forests for the same type of problem claiming that this algorithm outperforms some of the earlier mentioned algorithms on several problems [10].Yet another study applies an unsupervised feature selection algorithm and clustering to classify unlabeled documents [11].

## 4. Experiments

Our hypothesis is that it is possible to detect from the EULA whether the associated software contains spyware or not, i.e., if it should be classified as good or bad. To test this hypothesis we gather examples of good and bad software applications and their corresponding EULAs and generate a data set of instances represented by pairs of EULAs and application classifications. We then mine the generated data set using a set of popular algorithms to investigate whether the hypothesis should be rejected or accepted.

### 4.1 Data representation

Since the stated classification problem is quite analogous to that of classifying E-mail messages as either spam or legitimate, we choose to adopt a simple, yet successful way of representing the data from that area of research; we represent each EULA using a word frequency vector, thus the data instances are essentially represented by pairs of word frequency vectors and classes.

### 4.2 Data gathering process and classification

In order to get the best settings for testing the hypothesis the following search strategy was adopted when collecting applications to include in the data set; the applications should be easily downloaded from the Internet and they should present the user with a EULA that could be copied and pasted as ASCII text. The good software instances were collected by downloading the 50 most popular Windows applications from Download.com [25] and the bad applications were collected from SpywareGuide.com [26]. After the installation of each application the operating system was scanned with Adaware to verify the classification [29]. This verification showed that all applications associated with bad EULAs were detected by Adware, while no hits were found for the legitimate applications.

### 4.3 Data set format and pre-processing

We stored the data set using the Weka ARFF format in which each word frequency is represented by a numeric attribute and the class is represented by one nominal attribute (with two possible values; good or bad) [4]. The data set features 50 instances classified as good and 50 instances classified as bad, thus we did not have to deal with problems associated with a skewed class distribution. However, we believe that an equal class distribution will be difficult to achieve when creating larger data sets due to the simple fact that it is much harder to find bad applications and their corresponding EULAs. It could of course be argued that one should try to achieve a distribution that is close to the real-world distribution. Even though it is problematic to estimate this distribution it is commonly perceived that the amount of good software greatly outnumbers the amount of bad software. The word frequency vector was generated using Weka's StringToWordVector filter with the settings adjusted as in the study by Frank and Bouckaert [14], thus the TF IDF weight was applied, all characters were converted to lowercase, only alphabetic tokens were considered, stop words and hapax legomena were removed. We furthermore employed a form of feature selection in that we used the Weka default setting of storing a maximum of 1000 words per class to generate the data set.

### 4.4 Algorithms and settings

The main objective of this paper is not to determine the most suitable miner for the studied problem, which would most certainly involve extensive parameter tuning of each featured algorithm, but rather to determine if the hypothesis stated in the beginning of Section 4 should be accepted or rejected. To maximize the probability of finding a pattern, if indeed such a pattern exists at all, we chose to include a diverse population of 15 algorithms from different learning categories (e.g. functions, lazy learners, Bayesian learners, trees, meta-learners, rules, etc.). We used algorithm

implementations from Weka version 3.5.5 and applied the default configuration for each algorithm.

### 4.5 Experimental setup

The primary priority, in setting up the experiment, was that we needed to be able to test our hypothesis. We therefore needed to assess the accuracy of our candidates. Since we had a limited amount of data for training and testing (100 instances), we chose to perform repeated holdout tests to estimate prediction accuracy using two metrics; accuracy (correctly classified instances divided by total number of classified instances), and the area under the ROC curve (AUC). These metrics are by far the most widely used although one should keep in mind that there are issues both regarding accuracy and AUC as with most other metrics [2][29]. We note that many studies have shown the applicability of AUC for a wide range of data mining and machine learning problems, cf. Provost and Fawcett [3]. Intuitively, if our hypothesis holds, it should be possible to generate a classifier that should perform better on average than randomly guessing the class. Hence, we formulate the hypothesis test as follows; if anyone of the featured algorithms is significantly better than a random guesser on the featured data set for both accuracy and AUC we accept the hypothesis, otherwise we reject it. To investigate which algorithms performed significantly better than a random guesser we used repeated holdouts and the corrected paired t- test, which is a common combination used in similar applications [31]. We calculated the mean and standard deviation of 10 repeated holdouts with a 66% training set / 34% test set randomized split for each of the following metrics; accuracy (percent correct), AUC (including true positives rate and false positives rate), training time, and testing time. We used the corrected paired t-test (confidence 0.05, two-tailed) to compare each featured algorithm with a Weka baseline classifier called ZeroR, which classifies all instances as belonging to the same class, thus it shares the same results for both accuracy and AUC with a random guesser for a Boolean problem. We also compared the performance, in terms of accuracy, of the 15 featured algorithms with the state-of-the-art EULA analyzer tool according to the following procedure; we generated ten folds for testing by sampling, without replacement, 17 bad instances and 17 good instances for each fold (since the holdout procedure used to evaluate the 15 algorithms uses a 66/34 split) from the collection of EULAs. Obviously we did not generate any training folds since the EULA analyzer is a static model (in the sense that it does not learn). Since the EULA analyzer works by looking for keywords in plain text we used text document instances instead of word vector representations (which have been subjected to feature selection, etc.), thus it is important to keep this difference in mind when comparing the results later. More importantly, one should recognize that in a real-world scenario, the accuracy of EULA analyzer is dependent of the interpretation capabilities of the user concerning the resulting credit score for a particular EULA. For our experiment we used the optimal credit score cut-point which means that the published accuracy results of the EULA analyzer are likely to be higher than what can be achieved by the average user of the product. The process of submitting a EULA document to the analyzer and getting a classification is described in Section 2.3. We also emphasize that, since the testing folds for the EULA analyzer and the testing folds for the algorithms are not identical, we used a corrected non-paired t-test (confidence 0.05, two-tailed) for this part of the experiment. The objective was to find out for which algorithms there are significant improvements or degradations in performance compared to the state-of-the-art, however, as clearly mentioned earlier, we did not try to tune any of the learning algorithms to maximize performance (i.e., increasing the probability of finding significant improvements over the state-of-the-art). A secondary priority was to measure performance in terms of training and testing time. These priorities coincide with the objective of investigating if indeed a tool can be designed that builds upon the classification method presented in this paper to prevent the covert installation of spyware.

### 4.6 Experimental results

We present the results concerning the area under the ROC curve (AUC) and the accuracy in Table 1. It is clear that our hypothesis should be accepted since at least one classifier achieves a significant improvement, with regard to both AUC and accuracy, in comparison to the baseline classifier. There are, in fact, significant improvements of both accuracy and AUC for 13 out of 15 featured algorithms, excluding DecisionStump and Ridor. Moreover, when comparing the accuracy of the state-of-the-art EULA analysis method (the EULA analyzer) with the accuracy achieved by the featured algorithms it is also shown that 10 algorithms outperform this method, at least for the studied data set. However, only the improvements of Multinomial Naive Bayes and Support Vector Machines are statistically significant. For one algorithm, KStar, the accuracy is significantly degraded in comparison to the state-of-the-art. The high false positive rate of Kstar (0.77) might be alarming. However, it is important to recognize that our study merely features 100 instances. As more data is gathered for future work our hypothesis is that the performance will increase even for the worst performing algorithms. We further observe that Multinomial Naive Bayes is the best performing algorithm on this data set, achieving the best AUC and accuracy followed by Support Vector Machines, and Voted Perceptron. Support Vector Machines has the slowest training time out of these three candidates; however the testing time does not differ

significantly between them. Only KStar stands out, with regards to the measured testing time, with a mean result of approximately 10 seconds, while the other algorithms achieve results close to 0 seconds. Regarding training time, there is no algorithm needing more than 5 seconds and, in particular, HyperPipes, IBk, KStar, Multinomial Naive Bayes, and VotedPerceptron needs close to 0 seconds. Exactly what words the best behaving algorithms used for distinguishing between good and bad software EULAs could not be easily grasped because of their implicit representation of the learned classifiers. However, in our limited data set of 100 instances, tree and rule based algorithms identified single words such as "search" or "advertisements" for distinguishing between good and bad EULAs.

## 5. Discussion

The studied problem in general and the results from the conducted experiments in particular, raises several interesting issues which will now be addressed. We first bring forth some technical aspects related to the featured algorithms and their performance on EULA classification, and continue discussing the importance of automatic EULA analysis. This is followed by a proposal of a novel software tool for spyware prevention. Finally, we discuss what can be referred to as an arms race between E-mail spammers, spyware and malware vendors on one side and the countermeasure application developers on the other side.

### 5.1 Algorithm comparison

The results pertaining to the performance of the top three candidates for solving the studied problem seem to be well-aligned with results in related work; Multinomial Naive Bayes is known to perform very well on large vocabularies, i.e., when mining text documents that contain a large number of words [27]. However, it is usually acknowledged that Support Vector Machines outperforms Multinomial Naive Bayes on many problems, cf. [28]. Still, it should be considered that Multinomial Naive Bayes has no parameters that need to be tuned for a particular problem, while Support Vector Machines implementation does have a large number of configuration parameters, for instance the complexity constant, the choice of kernel, and the specific properties of the selected kernel. This would favor the Multinomial Naive Bayes algorithm in this study since only default configurations are used.

### 5.2 The relevance of EULA analysis

It is typically very hard for the average user to know if an application hosts spyware or not. The obvious way to gain such knowledge is to read the EULA of the application about to be installed. The distributors of software which include spyware programs specify the occurrence thereof in the EULAs that precede installation to avoid legal repercussions. Since users often find EULAs too lengthy to read and too complicated to understand, they neglect to inspect the EULAs and instead accept them and install the application unaware of the hidden spyware programs bound to infiltrate their operating systems [32]. As an example, one software vendor offered $1000 in prize money to the first person that contacts the company after reading the statement included in the EULA [33]. It took more than four months and over 3,000 downloads of the software before the prize money was claimed. A EULA for an application which is classified as good typically contains several thousand words [33]. However, EULAs for applications classified as bad are usually larger and more complex (as they often are intentionally written in a perplex way) and it is not uncommon for them to include warranties that minimize the vendors' responsibility and limit the user's access to the original code, e.g., by specifying that it is prohibited to reverse engineer the application or to eavesdrop on network packets sent from it [21]. It is evident that many users would benefit from using an automated tool, which can assist them in analyzing the contents of a EULA and predicting if the related software hosts spyware or not.

### 5.3 Towards a tool for spyware prevention

The EULA classification method outlined in this paper can be implemented as a software tool for spyware prevention. This tool should be designed as a middleware that operates between the operating system and the application bound to be installed. The tool should be executed as a background process set to identify and analyze a EULA as soon as it appears on the screen during an installation. Based on the result from the EULA analysis, the tool will provide the user with recommendations about the classification of the application. This allows the tool to assist users in making informed decisions about the installation of software without forcing them to read (and understand) the lengthy and intricate EULAs. Should a EULA be classified as bad, a user can take appropriate actions against it, e.g., by disagreeing with the EULA and exiting the installation process. It should however be noted that any tool based on our method should not be used in isolation, but in combination with other approaches, e.g., anti-spyware software (e.g. [29]). A similar application already exists, however, the significant differences lie in our use of data

mining algorithms (which has been shown in this study to be more accurate than keyword spotting services) on the one hand and the visualization of which parts of the EULA which significantly contributed to its classification as either good or bad software [36]. This visualization could, for instance, be implemented by using extracted rules or generated trees. If proven efficient, the tool should also be applicable in other settings, for instance in the case of privacy policies for Web sites and for the combination of privacy policies and EULAs. An interesting application would be a Web browser that automatically classifies privacy policies on Web sites visited and informs the user about the privacy status of those sites.

### 5.4 An arms race

Historically, there have been arms races between the distributors of spam e-mail messages and malware programs and those combating them. So far, there are some resemblances between this phenomenon and that of the spyware industry, for instance with respect to the similarities in design between anti-spyware and anti-malware applications. However, whereas malware and spam distributors really do not need to care for legal matters in licenses, etc., spyware vendors do. Spyware can only exist if their host applications are freely downloaded and installed by users wanting to gain access to the software host. For this purpose the EULAs are needed by the spyware industry. Moreover, this substantiates the significance of EULAs in the enhancement of user privacy. If spyware components were to be included without any references thereof in the EULA, the vendors of the host application face great risk of being brought to justice. Furthermore, the EULAs need to be constructed in a way so that they are valid and hold in court. We take advantage of this particular fact when mining the EULAs for patterns that separate good applications from bad.

## 6. Conclusions and Future Work

We investigate the possibility to predict whether a software application is, or hosts, spyware on the basis of its End User License Agreement (EULA). We conduct this investigation by collecting and analyzing 100 software applications and their corresponding EULAs by installing each application to determine and scanning the operating system with anti-spyware software, if the particular application is bad or good, i.e., if it hosts or indeed is classified as spyware or not. Each EULA text document is transformed to a word frequency vector and, together with the classification; it constitutes an instance that can be used for training and testing classifiers. The generated data set, which includes 100 classified EULA instances is available for downloading [30]. We conduct an empirical experiment in which 15 mining algorithms, 1 baseline classifier, and a state-of-the-art EULA analysis tool are applied on the data set. The results reveal that 13 out of the 15 algorithms significantly outperformed the baseline classifier in terms of both AUC and accuracy. Moreover, 2 algorithms also significantly outperform the state-of-the-art EULA analysis method, the EULA analyzer in terms of predictive accuracy. Most notably, the Multinomial Naive Bayes, Support Vector Machines, and Voted Perceptron algorithms achieve the highest AUC and accuracy scores and these three algorithms also share a low false positives rate. Our main conclusion is that the results strongly support our hypothesis that EULAs can indeed be used as a basis for classifying the corresponding software as good or bad. Based on this conclusion and the low training and testing times of most algorithms, we also conclude that it would be quite possible to use the EULA classification method in a spyware prevention tool that classifies the EULA when it is shown to a user during an application installation. The result from such an analysis gives the user a recommendation about the legitimacy of the application before the installation continues as well as some type of visualization of what information in the EULA that triggered this classification.

As future work, there is intent to implement and release such a software tool. Another interesting direction for future work is to collect data enough for a larger version of the empirical experiment conducted in this study in order for us to verify our main hypothesis and the results obtained. Further, it would be interesting to tune the parameters of the investigated algorithms included in the proposed extended experiment, cf. Lavesson and Davidsson [1]. Such an experiment could also involve more mining and feature selection algorithms, computational linguistic methods, other EULA text document representation except for word frequency vectors, as well as an analysis and visualization of words that trigger classification into legitimate software and spyware, respectively. Moreover, it might be possible to extend the domains of the tool to also include privacy policies of Web sites visited.

## 7. REFERENCES


[1] Lavesson, N., Davidsson, P., " Quantifying the Impact of Learning Algorithm Parameter Tuning", In Proceedings of the 21st AAAI National Conference on Artificial Intelligence (pp. 395-400), AAAI Press, Menlo Park, CA, 2006.

[2] Provost, F., T. Fawcett, and R. Kohavi, "The Case against Accuracy Estimation for Comparing Induction Algorithms", In Proceedings of the 15th International Conference on Machine Learning, pages 445-453, Morgan Kaufmann, San Francisco, CA, USA, 1998.

[3] Provost, F., and T. Fawcett, "Analysis and Visualization of Classifier Performance: Comparison under Imprecise Class

and Cost Distributions", In Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, pages 43-48, AAAI Press, Menlo Park, CA, USA, 1997.

[4] Witten, I. H., and E. Frank, Data Mining: Practical Machine Learning Tools and Techniques (2nd edition), Morgan Kaufmann, San Francisco, CA, USA, 2005.

[5] Cohen, W., "Learning Rules that Classify E-Mail", Advances in Inductive Logic Programming, IOS Press, Amsterdam, the Netherlands, 1996, pp. 124-143.

[6] Drucker, H., D. Wu, and V. Vapnik, "Support Vector Machines for Spam Categorization", IEEE Transactions on Neural Networks, 10(5), IEEE Press, New York City, NY, USA, 1999, pp. 1048-1054.

[7] Androutsopoulos, I., G. Paliouras, V. Karkaletsis, G. Sakkis, C. D. Spyropoulos, and P. Stamatopoulos, "Learning to Filter Spam E-Mail: A Comparison of a Naive Bayesian and a Memory-Based Approach", In Proceedings of the Workshop on Machine Learning and Textual Information Access, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases, pages 1-13, 2000.

[8] Carreras, X., and L. Marquez, "Boosting Trees for Anti-Spam Email Filtering", In Proceedings of the 4th International Conference on Recent Advances in Natural Language Processing, pages 58-64, 2001.

[9] Sakkis, G., I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. D. Spyropoulos and P. Stamatopoulos, "Stacking Classifiers for Anti-spam Filtering of E-mail", In Proceedings of the 6th Conference on Empirical Methods in Natural Language Processing, 2001.

[10] Koprinska, I., J. Poon, J. Clark, and J. Chan, "Learning to Classify E-Mail", Information Sciences, 177, Elsevier, Amsterdam, the Netherlands, 2007, pp. 2167-2187.

[11] Kang, N., C. Domeniconi, and D. Barbara, "Categorization and Keyword Identification of Unlabeled Documents", In Proceedings of the 5th IEEE International Conference on Data Mining, pages 677 - 680, IEEE Press, New York City, NY, USA, 2005.

[12] EULA Analyzer
http://www.spyw-areguide.com/analyze-
Last checked 2008-01-04.

[13] Saroiu, S., S. D. Gribble, and H. M. Levy, "Measurement and Analysis of Spyware in a University Environment", In Proceedings of the 1st Symposium on Networked Systems Design and Implementation, 2004.

[14] Frank, E., and R. R. Bouckaert, "Naive Bayes for Text Classification with Unbalanced Classes", In Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases, pages 503-510, Springer, Berlin, Germany, 2006.

[15] Boldt, M., Carlsson, B. and Jacobsson, A., "Exploring Spyware Effects", In Proceedings of the 8th Nordic Workshop on Secure IT Systems (NordSec04), Helsinki Finland, 2004.

[16] Jacobsson, A., Boldt, M., and Carlsson, B., "Privacy-Invasive Software in File-Sharing Tools", In Proceedings fo the 18th IFIP World Computer Congress, Toulouse France, 2004.

[17] McFedries, P., "The Spyware Nightmare", IEEE Spec-trum, 42(8), IEEE Press, New York City, NY, USA, 2005, pp. 72-72.

[18] Zhang, X., "What Do Consumers Really Know About Spyware?", Communications of the ACM, 48(8), ACM Press, New York City, NY, USA, 2005, pp. 44-48.

[19] Fox, S., "Spyware – The Threat of Unwanted Software Programs is Changing the Way People use the Internet", Pew Internet and American Life Project, 2005.

http://www.pewinternet.org/pdfs/PIP_Spyware_Report_July_05.pdf

Last checked 2008-01-04.

[20] Robertsson, B., "Five Major Categories of Spyware", Consumer Web Watch, October 21st, 2002.
http://www.consumerwebwatch.org/dynamic/

privacy-investigations-categories-spy.cfm

Last checked: 2008-01-04.

[21] Good, N., Grossklags, J., Thaw, D., Perzanowski, A., Mulligan, D.K., and Konstan, J., "User Choices and Regret: Understanding Users' Decision Process about Consensually Acquired Spyware", I/S Law and Policy for the Information Society, 2(2), ISJLP, Columbus, OH, USA, 2006, pp. 283-344.

[22] McCardle, M., "How Spyware Fits into Defence in Depth", SANS Reading Room, SANS Institute, 2003.

http://www.sans.org/rr/papers/index.php?id=905

Last checked 2008-01-04.

[23] Townsend, K., "Spyware, Adware, and Peer-to-Peer Net-works: The Hidden Threat to Corporate Security", PestPatrol, 2003.
http://www.pestpatrol.com/Whitepapers/CorporateSecurity0403.asp

Last checked 2008-01-04.

[24] Boldt, M., "Privacy-Invasive Software - Exploring Effects and Countermeasures", Licentiate Thesis Series No. 2007:01, School of Engineering, Blekinge Institute of Technology, Sweden, 2007.

[25] CNET Download.com
http://www.download.com
Last checked 2008-01-04.

[26] Spyware Guide
http://www.SpywareGuide.com
Last checked 2008-01-04.

[27] McCallum, A., and K. Nigam, "A Comparison of Event Models for Naive Bayes Text Classification", In Proceedings of the AAAI98 Workshop on Learning for Text Categorization, pages 41-48, Technical Report WS-98-05, AAAI Press, Menlo Park, CA, USA, 1998.

[28] Kibriya, A. M., E. Frank, B. Pfahringer, and G. Holmes, "Multinomial Naive Bayes for Text Categorization Revisited", In Proceedings of the 7th Australian Joint

Conference on Artificial Intelligence, pages 488-499, Springer, Heidelberg, Germany, 2004.

[29] Adaware
http://www.lavasoft.com
Last checked 2008-01-04.

[30] Niklas Lavesson, Blekinge Institute of Technology, 2007.
http://www.bth.se/tek/nla
Last accessed 2008-01-04.

[31] Nadeau, C., and Y. Bengio, "Inference for the Generalization Error", Machine Learning, 52(3), Springer, Amsterdam, the Netherlands, 2003, pp. 239–281.

[32] Sipior, J. C., B. T. Ward, and G. R. Roselli, "A United States Perspective on the Ethical and Legal Issues of Spyware", In Proceedings of the 7th International Conference on Electronic Commerce, pages 738 - 743, ACM Press, New York City, NY, USA, 2005.

[33] Good, N., R. Dhamija, J. Grossklags, D. Thaw, S. Aronowitz, D. Mulligan, and J. Konstan. "Stopping Spyware at the Gate: A User of Privacy, Notice and Spyware", In Proceedings of the Symposium on Usable Privacy and Security, pages 43-52, ACM Press, New York City, NY, USA, 2005.

[34] State of Spyware Q2 2006 – A Review and Analysis of the Impact of Spyware on Consumers and Corporations, Webroot Software, Inc., 2006.
http://h30307.www3.hp.com/pdf/SOS_Q206_USA.pdf
Last accessed 2008-01-04.

[35] State of Spyware Q1 2005 – A Review and Analysis of the Impact of Spyware on Consumers and Corporations, Webroot Software, Inc., 2006.
http://whitepapers.zdnet.co.uk/0,1000000651,260134901p,00.htm
Last accessed 2008-01-04.

[36] EULAlyzer
http://www.javacoolsoftware.com/eulalyzerpro.html
Last checked 2008-01-04.

## 8. Appendix

**Table 1. Predictive accuracies and area under the ROC curve results for 15 algorithms compared using a corrected paired t-test (confidence 0.05, two-tailed) with a baseline classifier which has the same performance as a random guesser. Significant improvements, compared to the baseline classifier, are shown with ●. True positives rates (TPR), false positives rates (FPR), training times, and testing times are also presented for each algorithm. All results are presented with the mean and standard deviation of 10 runs of holdout using a 66% training set / 34% test set randomized split. All algorithms are also compared, in terms of accuracy, using a regular, non-paired corrected t-test (confidence 0.05, two-tailed) with the state-of-the-art EULA analyzer. Significant improvements, compared to the EULA analyzer, are shown with + while significant degradations are shown with ○.**

| Algorithm | Accuracy % correct | TPR (TP/TP+FN) | FPR (FP/FP+TN) | AUC | Training Time seconds | Testing Time seconds |
|---|---|---|---|---|---|---|
| AdaBoostM1 | 73.82(5.79) ● | 0.72(0.08) | 0.24(0.09) | 0.78(0.04) ● | 3.55(0.28) | 0.00(0.01) |
| DecisionStump | 68.82(11.11) | 0.54(0.16) | 0.16(0.19) | 0.69(0.11) | 0.33(0.08) | 0.00(0.00) |
| HyperPipes | 76.47(7.59) ● | 0.91(0.14) | 0.38(0.19) | 0.90(0.07) ● | 0.04(0.01) | 0.07(0.09) |
| IBk | 77.94(5.59) ● | 0.71(0.07) | 0.15(0.10) | 0.78(0.06) ● | 0.04(0.01) | 0.13(0.02) |
| J48 | 73.24(10.23) ● | 0.72(0.16) | 0.26(0.18) | 0.73(0.10) ● | 1.29(0.23) | 0.00(0.01) |
| JRip | 71.18(5.33) ● | 0.71(0.14) | 0.29(0.12) | 0.72(0.07) ● | 2.02(0.23) | 0.00(0.00) |
| KStar | 59.71(4.17) ●○ | 0.96(0.03) | 0.77(0.09) | 0.68(0.07) ● | 0.00(0.00) | 9.20(0.42) |
| NaiveBayes | 79.41(9.80) ● | 0.91(0.09) | 0.32(0.17) | 0.80(0.10) ● | 0.31(0.02) | 0.11(0.05) |
| NaiveBayesNominal | 87.94(6.42) ●+ | 0.88(0.11) | 0.12(0.11) | 0.93(0.06) ● | 0.03(0.01) | 0.00(0.01) |
| PART | 72.65(10.74) ● | 0.72(0.15) | 0.26(0.15) | 0.72(0.11) ● | 2.41(2.15) | 0.00(0.01) |
| RandomForest | 75.29(7.10) ● | 0.79(0.09) | 0.28(0.09) | 0.83(0.08) ● | 3.64(0.20) | 0.00(0.00) |
| RBFNetwork | 77.35(7.73) ● | 0.75(0.12) | 0.21(0.13) | 0.78(0.09) ● | 1.46(0.19) | 0.17(0.02) |
| Ridor | 67.65(11.35) | 0.63(0.14) | 0.28(0.13) | 0.68(0.11) | 0.87(0.11) | 0.00(0.01) |
| SMO | 83.53(3.97) ●+ | 0.78(0.08) | 0.11(0.08) | 0.84(0.04) ● | 0.25(0.08) | 0.00(0.00) |
| VotedPerceptron | 81.47(6.66) ● | 0.85(0.13) | 0.22(0.10) | 0.87(0.07) ● | 0.04(0.01) | 0.02(0.01) |
| ZeroR (baseline) | 50.00(0.00) | 1.00(0.00) | 1.00(0.00) | 0.50(0.00) | 0.00(0.01) | 0.00(0.00) |
| EULA analyzer | 72.7 (3.86) | N/A | N/A | N/A | N/A | N/A |