Is Feature Selection Secure against Training Data Poisoning?

Huang Xiao\textsuperscript{2}, Battista Biggio\textsuperscript{1}, Gavin Brown\textsuperscript{3}, Giorgio Fumera\textsuperscript{1}, Claudia Eckert\textsuperscript{2}, Fabio Roli\textsuperscript{1}

\textsuperscript{(1)} Dept. Of Electrical and Electronic Engineering, University of Cagliari, Italy
\textsuperscript{(2)} Department of Computer Science, Technische Universität München, Germany
\textsuperscript{(3)} School of Computer Science, University of Manchester, UK
Motivation

• Increasing number of services and apps available on the Internet
  – Improved user experience

• Proliferation and sophistication of attacks and cyberthreats
  – Skilled / economically-motivated attackers

• Several security systems use machine learning to detect attacks
  – but ... is machine learning secure enough?
Is Feature Selection Secure?

- **Adversarial ML**: security of *learning and clustering* algorithms
  - Barreno et al., 2006; Huang et al., 2011; Biggio et al., 2014; 2012; 2013a; Brueckner et al., 2012; Globerson & Roweis, 2006

- **Feature Selection**
  - High-dimensional feature spaces (e.g., spam and malware detection)
  - Dimensionality reduction to improve interpretability and generalization

\[
\begin{pmatrix}
  x_1 \\
  x_2 \\
  ... \\
  ... \\
  x_d
\end{pmatrix}
\rightarrow
\begin{pmatrix}
  x_{(1)} \\
  x_{(2)} \\
  ... \\
  x_{(k)}
\end{pmatrix}
\]

- How about the **security** of feature selection?
Feature Selection under Attack

Attacker Model

- **Goal** of the attack
- **Knowledge** of the attacked system
- **Capability** of manipulating data
- **Attack strategy**
Attacker’s Goal

- **Integrity Violation**: to perform malicious activities without compromising normal system operation
  - enforcing selection of features to facilitate evasion at test time

- **Availability Violation**: to compromise normal system operation
  - enforcing selection of features to maximize generalization error

- **Privacy Violation**: gaining confidential information on system users
  - reverse-engineering feature selection to get confidential information
Attacker’s Knowledge

- **Perfect knowledge**
  - upper bound on performance degradation under attack

- **Limited knowledge**
  - attack on surrogate data sampled from same distribution
Attacker’s Capability

- **Inject points** into the **training** data

- **Constraints on data manipulation**
  - Fraction of the training data under the attacker’s control
  - Application-specific constraints

- **Example on PDF data**
  - PDF file: hierarchy of interconnected objects
  - Objects can be added but not easily removed without compromising the file structure

```
13 0 obj
<< /Kids [ 1 0 R 11 0 R ]
/Type /Page
... >> end obj
```

```
17 0 obj
<< /Type /Encoding
/Differences [ 0 /C0032 ] >> endobj
```
Attack Scenarios

• Different potential attack scenarios depending on assumptions on the attacker’s goal, knowledge, capability
  – Details and examples in the paper

• Poisoning Availability Attacks
  *Enforcing selection of features to maximize generalization error*
  – **Goal**: availability violation
  – **Knowledge**: perfect / limited
  – **Capability**: injecting samples into the training data
Embedded Feature Selection Algorithms

- **Linear models** \( f(x) = w^T x + b \)
  - Select features according to \(|w|\)

\[
\min_{w,b} \mathcal{L} = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)) + \lambda \Omega(w)
\]

\[
= \frac{1}{2} (f(x_i) - y_i)^2
\]

- **LASSO**
  - Tibshirani, 1996
- **Ridge Regression**
  - Hoerl & Kennard, 1970
- **Elastic Net**
  - Zou & Hastie, 2005
Poisoning Embedded Feature Selection

- **Attacker’s objective**
  - to maximize generalization error on untainted data

\[
\max_{\mathbf{x}_c} \mathcal{W} = \frac{1}{m} \sum_{j=1}^{m} \ell(\hat{y}_j, f(\hat{x}_j)) + \lambda \Omega(\mathbf{w})
\]

... w.r.t. choice of the attack point

- **Solution:** subgradient-ascent technique

Loss estimated on surrogate data (excluding the attack point)

\[
\hat{\mathcal{D}} = \{(\hat{x}_i, \hat{y}_i) \}_{i=1}^{m}
\]

Algorithm is trained on surrogate data (including the attack point)

\[
\mathcal{L}(\hat{\mathcal{D}} \cup \{\mathbf{x}_c\})
\]
Gradient Computation

$$\frac{\partial W}{\partial x_c} = \frac{1}{m} \sum_{j=1}^{m} (f(\hat{x}_j) - \hat{y}_j) \left( x_j^\top \frac{\partial w}{\partial x_c} + \frac{\partial b}{\partial x_c} \right) + \lambda \frac{\partial \Omega}{\partial w} \frac{\partial w}{\partial x_c}$$

How does the solution change w.r.t. $x_c$?

KKT conditions

$$\frac{\partial \mathcal{L}^\top}{\partial w} = \frac{1}{m+1} \sum_{j=1}^{m+1} (f(\hat{x}_j) - \hat{y}_j) \hat{x}_j + \lambda \frac{\partial \Omega}{\partial w}^\top = 0$$

$$\frac{\partial \mathcal{L}}{\partial b} = \frac{1}{m+1} \sum_{j=1}^{m+1} (f(\hat{x}_j) - \hat{y}_j) = 0$$

Subgradient is unique at the optimal solution!

$$\frac{\partial \Omega}{\partial w} = -\frac{1}{\lambda} \frac{1}{m+1} \sum_{j=1}^{m+1} (f(\hat{x}_j) - \hat{y}_j) \hat{x}_i^\top$$
Gradient Computation

- We require the KKT conditions to hold under perturbation of $x_c$

\[
\begin{bmatrix}
\Sigma + \lambda v & \mu \\
\mu^\top & 1
\end{bmatrix}
\begin{bmatrix}
\frac{\partial w}{\partial x_c} \\
\frac{\partial b}{\partial x_c}
\end{bmatrix}
= -\frac{1}{m + 1}
\begin{bmatrix}
M \\
w^\top
\end{bmatrix}
\]

\[
\frac{\partial W}{\partial x_c} = \frac{1}{m} \sum_{j=1}^{m} (f(\hat{x}_j) - \hat{y}_j) \left( \hat{x}_j^\top \frac{\partial w}{\partial x_c} + \frac{\partial b}{\partial x_c} \right) + \lambda \frac{\partial \Omega}{\partial w} \frac{\partial w}{\partial x_c}
\]

Gradient is now uniquely determined
Poisoning Attack Algorithm

**Algorithm 1** Poisoning Embedded Feature Selection

**Input:** \( \hat{\mathcal{D}} \), the (surrogate) training data; \( \{x_c^{(0)}, y_c\}_{c=1}^q \), the \( q \) initial attack points with (given) labels; \( \beta \in (0, 1) \); and \( \sigma, \varepsilon \), two small positive constants.

**Output:** \( \{x_c\}_{c=1}^q \), the final attack points.

1: \( p \leftarrow 0 \)
2: **repeat**
3:   **for** \( c = 1, \ldots, q \) **do**
4:     \( \{w, b\} \leftarrow \text{learn the classifier on } \hat{\mathcal{D}} \cup \{x_c^{(p)}\}_{c=1}^q \).
5:     Compute \( \nabla \mathcal{W} = \frac{\partial \mathcal{W}(x_c^{(p)})}{\partial x_c} \) according to Eq. (4).
6:     Set \( d = \Pi_B \left( x_c^{(p)} + \nabla \mathcal{W} \right) - x_c^{(p)} \) and \( k \leftarrow 0 \).
7:     **repeat** \{line search to set the gradient step \( \eta \} \)
8:       Set \( \eta \leftarrow \beta^k \) and \( k \leftarrow k + 1 \)
9:       \( x_c^{(p+1)} \leftarrow x_c^{(p)} + \eta d \)
10:      **until** \( \mathcal{W}(x_c^{(p+1)}) \leq \mathcal{W}(x_c^{(p)}) - \sigma \eta \|d\|^2 \)
11:   **end for**
12: \( p \leftarrow p + 1 \)
13: **until** \( \|\mathcal{W}(\{x_c^{(p)}\}_{c=1}^q) - \mathcal{W}(\{x_c^{(p-1)}\}_{c=1}^q)\| < \varepsilon \)
14: **return:** \( \{x_c\}_{c=1}^q = \{x_c^{(p)}\}_{c=1}^q \)
Experiments on PDF Malware Detection

- **PDF**: hierarchy of interconnected objects (keyword/value pairs)

  13 0 obj
  << /Kids [ 1 0 R 11 0 R ]
  /Type /Page
  ... >> end obj

  17 0 obj
  << /Type /Encoding
  /Differences [ 0 /C0032 ] >>
  endobj

- **Features**: keyword counts

<table>
<thead>
<tr>
<th>Feature</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>/Type</td>
<td>2</td>
</tr>
<tr>
<td>/Page</td>
<td>1</td>
</tr>
<tr>
<td>/Encoding</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Learner’s task**: to classify *benign* vs *malware* PDF files
- **Attacker’s task**: to maximize classification error by injecting poisoning attack samples
  - Only feature increments are considered (object insertion)
    - Object removal may compromise the PDF file

Maiorca et al., 2012; 2013; Smutz & Stavrou, 2012; Srndic & Laskov, 2013
Experimental Results

Data: 300 (TR) and 5,000 (TS) samples – 114 features

Similar results obtained for limited-knowledge attacks!
Experimental Results

$IC(A, B) = \frac{rd - k^2}{k(d - k)} \in [-1, +1]$

Kuncheva et al., 2007

| A: selected features in the absence of attack |
| B: selected features under attack |
| k: number of features selected out of d |
| r: common features between the two sets |
Conclusions and Future Work

• Framework for **security evaluation** of **feature selection** under attack
  – Poisoning attacks against embedded feature selection algorithms

• Poisoning can significantly affect feature selection
  – LASSO significantly vulnerable to poisoning attacks

  **L1 regularization:** stability against random noise, but not against adversarial (worst-case) noise?

• **Future research directions**
  – Error bounds on the impact of poisoning on learning algorithms
  – Secure / robust feature selection algorithms
Thanks for your attention!

Any questions?
Experimental Results

Classification Error

Feature Subset Size

Feature Stability (k=30)

Feature Stability (k=50)

Perfect Knowledge

Limited Knowledge

http://pralab.diee.unica.it