Adversarial Regression with Multiple Learners

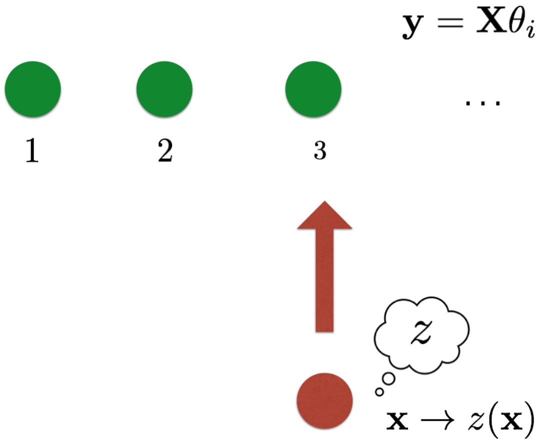
Liang Tong*[1] Sixie Yu*[1] Scott Alfeld[2] Yevgeniy Vorobeychik[1]

[1]Electrical Engineering and Computer Science
Vanderbilt University

[2]Computer Science
Amherst College

ICML 2018

Problem Setting



Motivation

- Adversaries can change features at test time to cause incorrect predictions.
  - i.e., change features of a house (i.e., square feet, #rooms) to fool online real-estate evaluation system, or make invisible changes to pictures to fool classifier.

Motivation

- Adversaries can change features at test time to cause incorrect predictions.
  - i.e., change features of a house (i.e., square feet, #rooms) to fool online real-estate evaluation system, or make invisible changes to pictures to fool classifier.
- Previous investigations of this problem pit a single learner against an adversary. [**Bruckner11**, **Dalvi04**, **li2014feature**, **zhou2012**]

Motivation

- Adversaries can change features at test time to cause incorrect predictions.
  - i.e., change features of a house (i.e., square feet, #rooms) to fool online real-estate evaluation system, or make invisible changes to pictures to fool classifier.
- Previous investigations of this problem pit a single learner against an adversary. [**Bruckner11**, **Dalvi04**, **li2014feature**, **zhou2012**]
- But an adversary's decision is usually aimed at a collection of learners.
  - i.e., an adversary crafts generic malwares and disseminate them widely.

Motivation

- Adversaries can change features at test time to cause incorrect predictions.
  - i.e., change features of a house (i.e., square feet, #rooms) to fool online real-estate evaluation system, or make invisible changes to pictures to fool classifier.
- Previous investigations of this problem pit a single learner against an adversary. [**Bruckner11**, **Dalvi04**, **li2014feature**, **zhou2012**]
- But an adversary's decision is usually aimed at a collection of learners.
  - i.e., an adversary crafts generic malwares and disseminate them widely.
- The learners all make autonomous decisions about how to detect malicious content.

Table of Contents

Learner Model

- $(\mathbf{X}, \mathbf{y})$: training dataset from an unknown distribution $\mathcal{D}$.
- $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_m]^\top$ and $\mathbf{y} = [y_1, y_2, ..., y_m]^\top$: $\mathbf{x}_j$ the jth instance and y_j its corresponding response variable.
- Test data is drawn from a distribution $\mathcal{D}^{'}$ (a modification of $\mathcal{D}$) manipulated by the attacker.
- An instance from $\mathcal{D}^{'}$ ($\mathcal{D}$) with probability β $(1-\beta)$.
- The action of the ith learner is to learn the parameters of the linear regression model: $\boldsymbol{\theta}_i$, which results in $\hat{\mathbf{y}}_i = \mathbf{X}\boldsymbol{\theta}_i$.

The expected cost function of the ith learner:

$$c_i(\boldsymbol{\theta}_i, \mathcal{D}^{'}) = \beta \mathbb{E}_{(\mathbf{X}^{'}, \mathbf{y}) \sim \mathcal{D}^{'}}[\ell(\mathbf{X}^{'}\boldsymbol{\theta}_i, \mathbf{y})] + (1-\beta)\mathbb{E}_{(\mathbf{X}, \mathbf{y}) \sim \mathcal{D}}[\ell(\mathbf{X}\boldsymbol{\theta}_i, \mathbf{y})] \quad (1)$$

where $\ell(\hat{\mathbf{y}}, \mathbf{y}) = ||\hat{\mathbf{y}} - \mathbf{y}||_2^2$.

Table of Contents

Attacker Model

- Every instance $(\mathbf{x}, y)$ is maliciously modified by the attacker to $(\mathbf{x}^{'}, y)$, with probability β.
- Assume the attacker has an instance-specific target $z(\mathbf{x})$.
- The objective of the attacker: $\hat{y} = \boldsymbol{\theta}_i^{\top}\mathbf{x}^{'}$ close to $z(\mathbf{x})$.
- The attacker's objective is measured by: $\ell(\hat{\mathbf{y}}, \mathbf{z}) = ||\hat{\mathbf{y}} - \mathbf{z}||_2^2$.
- Transforming $\mathbf{X}$ to $\mathbf{X}^{'}$ incurs costs: $R(\mathbf{X}^{'}, \mathbf{X}) = ||\mathbf{X}^{'} - \mathbf{X}||_F^2$.

The expected cost function of the attacker:

$$c_a(\{\boldsymbol{\theta}_i\}_{i=1}^n, \mathbf{X}^{'}) = \sum_{i=1}^{n} \ell(\mathbf{X}^{'}\boldsymbol{\theta}_i, \mathbf{z}) + \lambda R(\mathbf{X}^{'}, \mathbf{X}) \tag{2}$$

Table of Contents

Multi-Learner Stackelberg Game (MLSG)

- The MLSG has two stages, which proceeds as follow:
  - In the first stage the learners simultaneously learn their model parameters $\{\boldsymbol{\theta}_\mathbf{i}\}_{i=1}^n$.
  - In the second stage, *after observing the learners' decision*, the attacker constructs its optimal attack (manipulating $\mathbf{X}$).

Assumptions

- The learners have complete information about β, λ, and $\mathbf{z}$.
- Each learner has the same action space $\boldsymbol{\Theta} \subseteq \mathbb{R}^{d\times 1}$, which is nonempty, compact, and convex.
- The columns of the test data $\mathbf{X}$ are linearly independent.

Multi-Learner Stackelberg Game (MLSG)

Definition (Multi-Learner Stackelberg Equilibrium (MLSE))

An action profile $(\{\boldsymbol{\theta}_i^*\}_{i=1}^n, \mathbf{X}^*)$ is an MLSE if it satisfies

$$\begin{aligned} \boldsymbol{\theta}_i^* &= \arg\min_{\boldsymbol{\theta}_i \in \boldsymbol{\Theta}} c_i(\boldsymbol{\theta}_i, \mathbf{X}^*(\boldsymbol{\theta})), \forall i \in \mathcal{N} \\ \texttt{s.t.} \quad \mathbf{X}^*(\boldsymbol{\theta}) &= \arg\min_{\mathbf{X}' \in \mathbb{R}^{m \times d}} c_a(\{\boldsymbol{\theta}_i\}_{i=1}^n, \mathbf{X}'). \end{aligned} \tag{3}$$

where $\boldsymbol{\theta} = \{\boldsymbol{\theta}_i\}_{i=1}^n$ constitutes the joint actions of the learners.

- MLSE is a blend between a Nash equilibrium (among all learners) and a Stackelberg equilibrium (between the learners and the attacker).

Multi-Learner Stackelberg Game (MLSG)

Lemma (Best Response of the Attacker)

Given $\{\boldsymbol{\theta}_i\}_{i=1}^n$, the best response of the attacker is

$$\mathbf{X}^* = (\lambda\mathbf{X} + \mathbf{z}\sum_{i=1}^{n}\boldsymbol{\theta}_i^\top)(\lambda\mathbf{I} + \sum_{i=1}^{n}\boldsymbol{\theta}_i\boldsymbol{\theta}_i^\top)^{-1}. \quad (4)$$

- $\mathbf{X}^*$ has a closed form, as a function of $\{\boldsymbol{\theta}_i\}_{i=1}^n$.
- With this lemma, the learners' cost functions become:

$$c_i(\boldsymbol{\theta}_i, \boldsymbol{\theta}_{-i}) = \beta\ell(\mathbf{X}^*(\boldsymbol{\theta}_i, \boldsymbol{\theta}_{-i})\boldsymbol{\theta}_i, \mathbf{y}) + (1-\beta)\ell(\mathbf{X}\boldsymbol{\theta}_i, \mathbf{y}). \quad (5)$$

- MLSG $\xRightarrow{\mathbf{X}^*(\boldsymbol{\theta}_i, \boldsymbol{\theta}_{-i})}$ Multi-Learner Nash Game (MLNG)
- MLNG is a game among the learners.

Table of Contents

Existence and Uniqueness of the Equilibrium

We approximate the MLNG by deriving upper bounds on the learners' cost functions. The approximated game is denoted by: $\langle \mathcal{N}, \boldsymbol{\Theta}, (\widetilde{c}_i) \rangle$.

Theorem (Existence of Nash Equilibrium)

$\langle \mathcal{N}, \boldsymbol{\Theta}, (\widetilde{c}_i) \rangle$ *is a symmetric game and it has at least one symmetric equilibrium.*

Theorem (Uniqueness of Nash Equilibrium)

$\langle \mathcal{N}, \boldsymbol{\Theta}, (\widetilde{c}_i) \rangle$ *has an unique Nash equilibrium, and this unique NE is symmetric.*

The equilibrium of $\langle \mathcal{N}, \boldsymbol{\Theta}, (\widetilde{c}_i) \rangle$ is defined as: *Multi-Learner Nash Equilibrium (MLNE)*

Table of Contents

Computing the MLNE

By utilizing first-order optimality conditions of each learner's optimization problem:

Theorem

Let

$$f(\boldsymbol{\theta}) = \ell(\mathbf{X}\boldsymbol{\theta}, \mathbf{y}) + \frac{\beta(n+1)}{2\lambda^2}||\mathbf{z} - \mathbf{y}||_2^2(\boldsymbol{\theta}^\top \boldsymbol{\theta})^2, \tag{6}$$

Then, the unique symmetric NE of $\langle \mathcal{N}, \boldsymbol{\Theta}, (\widetilde{c}_i) \rangle$, $\{\boldsymbol{\theta}_i^*\}_{i=1}^n$, *can be derived by solving the following convex optimization problem*

$$\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} f(\boldsymbol{\theta}) \tag{7}$$

and then letting $\boldsymbol{\theta}_i^* = \boldsymbol{\theta}^*, \forall i \in \mathcal{N}$, *where* $\boldsymbol{\theta}^*$ *is the solution of Eq.* (7).

Table of Contents

Robustness analysis

A robust linear regression solves the following problem:

$$\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \max_{\triangle\in\mathcal{U}} ||\mathbf{y} - (\mathbf{X} + \triangle)\boldsymbol{\theta}||_2^2, \tag{8}$$

where the uncertainty set
$\mathcal{U} = \{\triangle \in \mathbb{R}^{m\times d} \,|\, \triangle^T\triangle = \mathbf{G} : |\mathbf{G}_{ij}| \leq c|\theta_i\theta_j| \;\; \forall i, j\}$, with
$c = \frac{\beta(n+1)}{2\lambda^2}||\mathbf{z} - \mathbf{y}||_2^2$.

Theorem

The optimal solution $\boldsymbol{\theta}^*$ of the problem in Eq. (7) is an optimal solution to the robust optimization problem in Eq. (8).

Our Contribution

- Fomally model the interaction between the learners and the attacker as a *Multi-Learner Stackelberg Game*.

Our Contribution

- Fomally model the interaction between the learners and the attacker as a *Multi-Learner Stackelberg Game*.
- Approximate this game by deriving upper bounds on the learners' loss functions.

Our Contribution

- Fomally model the interaction between the learners and the attacker as a *Multi-Learner Stackelberg Game*.
- Approximate this game by deriving upper bounds on the learners' loss functions.
- Show that there always exists a *unique* symmetric equilibrium of the approximated game.

Our Contribution

- Fomally model the interaction between the learners and the attacker as a *Multi-Learner Stackelberg Game*.
- Approximate this game by deriving upper bounds on the learners' loss functions.
- Show that there always exists a *unique* symmetric equilibrium of the approximated game.
- Theoretically and experimently show that the equilibrium of the approximated game is robust.

Thank you!

- Poster: Hall B #120
- Email: sixie.yu@vanderbilt.edu
- Homepage: sixie-yu.org

Table of Contents

References