Game Theory and Boosting

Mose Wintner

Fall 2017

1 Classification

Suppose that we are given a paired finite set of data *instances* and associated binary *labels* $(X_i, Y_i) \in X \times \{\pm 1\}$, for $i = 1, \dots, n$, where the X_i are independently drawn from a probability distribution D on X. Suppose we wish to devise a concise *hypothesis* relating the instances in X to their labels, i.e. a function

$$h : X \to \{\pm 1\}.$$

We want to choose h in such a way as to minimize the generalization error

$$\mathbb{E}_{X \sim D}(h(X) - Y)^2.$$

This is the *classification problem.*

2 Weak Learning and Boosting

A standard assumption often made in learning is the γ-*weak learnability assumption*, i.e. that we have a "weak learning" algorithm and $\gamma > 0$ such that for any probability distribution D on X, the algorithm produces a "weak" hypothesis h of error at most $\frac{1}{2} - \gamma$ with respect to D. Under this assumption we may implement the following successful *boosting* algorithm, known as AdaBoost:

Given: $(X_1, Y_1), \dots, (X_n, Y_n)$ where $X_i \in X$, $Y_i \in \{\pm 1\}$ for each $i = 1, \dots, n$
Initialize: $D_1(i) = \frac{1}{n}$ for $i = 1, \dots, n$.
For $t = 1, \dots, T$:

- Train weak learning algorithm using distribution D_t
- Obtain weak hypothesis $h_t : X \to \{\pm 1\}$.
- Goal: Select h_t to minimize the weighted error

$$\varepsilon_t = Pr_{j \sim D_t}(h_t(X_i) \neq Y_i) = \mathbb{E}(h_t(X_i) - Y_i)^2$$

- Set the *weight* $\alpha_t := \frac{1}{2} \ln \left(\frac{1 - \varepsilon_t}{\varepsilon_t} \right)$.
- Update for $i = 1, \dots, n$

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \exp\left(\alpha_t (-1)^{I\{h_t(X_i) = Y_i\}} \right) \\ &= \frac{D_t(i) \exp(-\alpha_t Y_i h_t(X_i))}{Z_t} \end{aligned}$$

  where Z_t is a normalization factor chosen to make D_{t+1} a distribution.

- Output the final hypothesis

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x) \right).$$

3 Game Theory

Two-player games may be generally represented as involving a row player (the *learner*) and a column player (the *environment*) simultaneously choosing rows and columns, respectively, of a a matrix $\mathbf{M}$. The matrix entry $\mathbf{M}(i,j)$ contains the *loss* for the learner for choosing row i if the environment chooses column j. Throughout, i will index rows and j will index the columns of $\mathbf{M}$.

One example of such a game is rock-paper-scissors, presented by

	rock	paper	scissors
rock	1/2	1	0
paper	0	1/2	1
scissors	1	0	1/2

The learner seeks to minimize her loss. Though the two players' losses may be independent in general, we restrict for simplicity to zero sum games, in which the column player's loss is the negative of the learner's. Therefore the environment plays to maximize the learner's loss.

4 Repeated Play

In the study of games, probabilistic play, i.e. *mixed strategy* is allowed; the learner seeks an optimal distribution P over the rows, and the environment seeks an optimal distribution Q over the columns. The learner's expected loss is $\mathbf{M}(P,Q) := P^T\mathbf{M}Q$. Given that the column player will choose column j, the row player's expected loss is

$$\sum_i P(i)\mathbf{M}(i,j) =: \mathbf{M}(P,j).$$

Similarly we will define

$$\sum_j \mathbf{M}(i,j)Q(j) =: \mathbf{M}(i,Q).$$

Suppose we play such a zero sum game repeatedly as the learner over *rounds* $t = 1,\ldots,T$, where in round t,

1. The learner chooses a *strategy* P_t, i.e. a probability distribution over the rows of $\mathbf{M}$.
2. The environment responds with strategy Q_t.
3. The learner is permitted to observe the losses $\mathbf{M}(i,Q_t)$.
4. The learner suffers loss $\mathbf{M}(P_t,Q_t)$.

We wish to help the learner minimize her cumulative loss

$$\sum_{t=1}^{T} \mathbf{M}(P_t,Q_t).$$

5 Von Neumann's Minmax Theorem

Consider the problem of choosing whether to play first or second in a zero sum game. We would expect the row player's loss to be no worse in playing second, since she possesses knowledge of the environment's choice of action. Put differently,

$$\max_Q \min_P \mathbf{M}(P,Q) \leq \min_P \max_Q \mathbf{M}(P,Q).$$

The content of the minmax theorem is that equality holds. The common value of both sides is called the *value* of the game, denoted here by v.

Put in different terms, the minmax theorem guarantees that the learner has a strategy P^* (a *minmax* strategy) such that regardless of the environment's strategy Q, her loss suffered, $\mathbf{M}(P^*,Q)$, will not exceed v. Symmetrically, there is a strategy Q^* (a *maxmin* strategy) such that regardless of the learner's strategy P, the environment can force a loss $\mathbf{M}(P,Q^*) \geq v$. For example, the optimal strategy for either player in rock-paper-scissors is to play each option with probability $\frac{1}{3}$.

6 Multiplicative Weights Algorithm (MW)

Begin as the learner with some initial mixed strategy P_1 and a weight $\eta > 0$. We iteratively update the strategy in round t by the rule

$$P_{t+1}(i) = \frac{P_t(i)\exp(-\eta \mathbf{M}(i, Q_t))}{Z_t},$$

where

$$Z_t = \sum_i P_t(i)\exp(-\eta \mathbf{M}(i, Q_t))$$

is a normalization factor. The following are results analyzing the performance of the MW algorithm.

Theorem 1 *For any matrix $\mathbf{M}$ with m rows and entries in $[0,1]$, and for any sequence of mixed strategies $Q_1, \dots, Q_T$ played by the environment, the sequence of mixed strategies $P_1, \dots, P_T$ prescribed by MW with parameter η satisfies*

$$\sum_{t=1}^{T} \mathbf{M}(P_t, Q_t) \leq \min_P \left[a_\eta \sum_{t=1}^{T} \mathbf{M}(P, Q_t) + c_\eta KL(P||P_1) \right],$$

where

$$a_\eta := \frac{\eta}{1 - e^{-\eta}}, \qquad c_\eta := \frac{1}{1 - e^{-\eta}},$$

and $KL(P||P')$ denotes the Kullback-Leibler divergence

$$KL(P||P') := \sum_{i=1}^{m} P(i) \ln \left(\frac{P(i)}{P'(i)} \right).$$

Corollary 1 *If MW is used with P_1 set equal to the uniform distribution, then its total loss is bounded by*

$$\sum_{t=1}^{T} \mathbf{M}(P_t, Q_t) \leq a_\eta \min_P \sum_{t=1}^{T} \mathbf{M}(P, Q_t) + c_\eta \ln m.$$

Corollary 2 *Under the conditions of the theorem and previous corollary and with*

$$\eta = \ln \left(1 + \sqrt{\frac{2 \ln m}{T}} \right),$$

the average per-trial loss suffered by the learner is

$$\frac{1}{T} \sum_{t=1}^{T} \mathbf{M}(P_t, Q_t) \leq \min_P \frac{1}{T} \sum_{t=1}^{T} \mathbf{M}(P, Q_t) + \Delta_T,$$

where

$$\Delta_T = \sqrt{\frac{2 \ln m}{T}} + \frac{\ln m}{T}.$$

In other words, MW yields a strategy which underperforms compared to the optimal strategy by an additive constant $= O\left(\sqrt{\frac{\ln m}{T}}\right) \to 0$ as $T \to \infty$.

7 Online Learning

In addition to training and testing being separate, typically the instances of data are assumed to be iid from some probability distribution. Suppose we are interested in *online learning*, where instead of separating the training and testing phases, one observes a sequence of instances and predicts their labels one by one. Additionally, *no* assumption of an underlying distribution is made; the instances observed may even be generated adversarially, so there may be no way to limit the number of mistakes. Therefore we search for an optimum in some reasonably limited set of hypotheses $\mathcal{H}$.

Assume that there are finitely many observable instances, indexed by $x \in X$, and finitely many hypotheses $\mathcal{H} = \{h : X \to \{\pm 1\}\}$. We also assume that there is some *target* $c : X \to \{\pm 1\}$, not necessarily in $\mathcal{H}$, which correctly classifies each instance $x \in X$ (we need not implicitly assume that the same instance never appears twice with different labels, but we will here). The goal of online learning is to minimize the number of mistakes, i.e. to find

$$\operatorname*{argmin}_{h \in \mathcal{H}} \sum_{t=1}^{T} I\{h(x_t) \neq c(x_t)\}.$$

8 Online Learning using MW

We may construct a repeated game playing problem and solve it using MW in order to solve the online learning problem. We construct our matrix $\mathbf{M} \in \{0,1\}^{|X| \times |\mathcal{H}|}$ as a *mistake matrix* with $\mathbf{M}(h, x) = 1$ if and only if $h(x) \neq c(x)$, i.e. if hypothesis h disagrees with the target on instance x. The expected number of mistakes made by the learner is therefore

$$\mathbb{E}\left[\sum_{t=1}^{T} I\{\hat{y}_t \neq c(x_t)\}\right] = \sum_{t=1}^{T} P(\hat{y}_t \neq c(x_t)).$$

We then apply MW to this matrix $\mathbf{M}$. The expected accuracy of the learner using MW will not exceed that of the best hypothesis in $\mathcal{H}$ by more than $O\left(\sqrt{\frac{\ln|\mathcal{H}|}{T}}\right)$.

What does the minmax theorem say about $\mathbf{M}$? Suppose its value is v. The minmax theorem tells us

$$\begin{aligned}
\min_P \max_{x \in X} \mathbf{M}(P, x) &= \min_P \max_Q \mathbf{M}(P, Q) \\
&= v \\
&= \max_Q \min_P \mathbf{M}(P, Q) \\
&= \max_Q \min_{h \in \mathcal{H}} \mathbf{M}(h, Q).
\end{aligned}$$

We have by definition that $\mathbf{M}(h, Q) = Pr_{x \sim Q}(h(x) \neq c(x))$. Therefore there exists a (maxmin) distribution Q^* on X such that for every hypothesis h,

$$\mathbf{M}(h, Q^*) = Pr_{x \sim Q^*}(h(x) \neq c(x)) \geq v.$$

On the other hand, γ-weak learnability tells us that there exists a hypothesis h depending on Q^* such that $Pr_{x \sim Q^*}(h(x) \neq c(x)) \leq \frac{1}{2} - \gamma$; therefore

$$v \leq \frac{1}{2} - \gamma.$$

The other piece of the theorem tells us that there exists a strategy P^* on $\mathcal{H}$ such that for all $x \in X$,

$$Pr_{h \sim P^*}(h(x) \neq c(x)) = \mathbf{M}(P^*, x) \leq v \leq \frac{1}{2} - \gamma < \frac{1}{2}.$$

Therefore by weighting each hypothesis h by $P^*(h)$ will correctly classify all instances of x:

$$c(x) = \operatorname{sign}\left(\sum_{h \in \mathcal{H}} P^*(h) h(x)\right).$$

9 Online Learning using Boosting

We can't apply a boosting algorithm like AdaBoost to the mistake matrix $\mathbf{M}$, because boosting generates a distribution over the rows, which in the case of $\mathbf{M}$ are the set of hypotheses; $\mathbf{M}$ has the instances as its columns. Therefore we would think to switch the row and column players. However, we can't just take

$\mathbf{M}^T$ because the learner of $\mathbf{M}^T$ (i.e. the environment of $\mathbf{M}$) wants to maximize the loss, but we want the row player of our game to minimize loss. Therefore instead we take the dual of the game, which we take to be $\mathbf{M}' := 1 - \mathbf{M}^T$ so its entries remain in $[0, 1]$. It has $|X|$ rows and $|\mathcal{H}|$ columns indexed by instances and hypotheses, respectively, and

$$\mathbf{M}'(x, h) = I\{h(x) = c(x)\}.$$

The most important elements of a boosting algorithm are choosing the distributions D_t and aggregating the weak hypotheses h_t into a final hypothesis. Note that any minmax strategy of $\mathbf{M}$ is a maxmin strategy of $\mathbf{M}'$, which means we're really searching for a maxmin strategy of $\mathbf{M}'$. We can design a boosting algorithm to approximate a maxmin strategy for $\mathbf{M}'$. After setting P_1 as in MW to be the uniform distribution over X, after each round of boosting $t = 1, \ldots, T$,

1. Set $D_t = P_t$ and pass D_t to the weak learning algorithm;
2. Get a hypothesis satisfying
$$Pr_{x \sim D_t}(h_t(x) = c(x)) \geq \frac{1}{2} + \gamma;$$
3. Let Q_t be the pure strategy concentrated on h_t and compute
$$\mathbf{M}'(x, Q_t) = I\{h_t(x) = c(x)\}$$
for each $x \in X$;
4. Update
$$P_{t+1}(x) = \frac{P_t(x)}{Z_t} e^{-\eta I\{h_t(x) = c(x)\}},$$
where as before Z_t is a normalization factor.

It can be shown that $\overline{Q} = \frac{1}{n} \sum_{t=1}^{T} Q_t$ is an approximate maxmin strategy. Since the strategy Q_t is focused on a single hypothesis h_t, we choose a final hypothesis H based on the simple majority

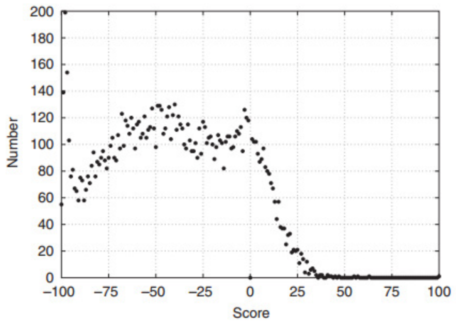
$$H(x) = \text{sign}\left(\sum_{t=1}^{T} h_t(x)\right).$$

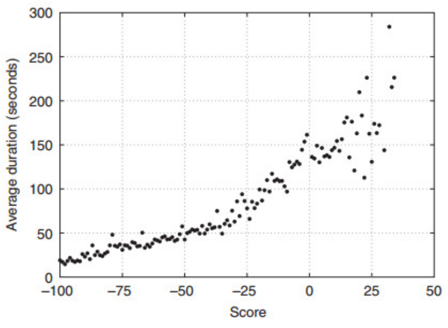
It can be shown that after $T = \Omega(\ln |X| / \gamma^2)$ rounds, the final output hypothesis is *equal* to the target $c(x)$, even if $c \notin \mathcal{H}$.

10 Test Case: Binary Pattern Recognition in Repeated Games

Consider a game where each of two players has an bit, sets it to 0 or 1 and conceals it. The players reveal simultaneously. Player 1 wins if the bits agree; otherwise player 2 wins. The optimal strategy in this game is, of course, to play either possibility uniformly at random. However, there is evidence to show that repeated play against humans in this game is not adversarial. In particular, David Hagelbarger and Claude Shannon designed a game-playing boosting algorithm to mine for human patterns in this game.

For example, suppose a human environment plays the sequence 0101010101... It would be natural for the computer learner to predict that the human will play 0 next. Such a pattern would therefore be picked up easily by an algorithm which made predictions based on past patterns. This is what Hagelbarger and Shannon implemented, using decision trees which specified predictions based on recent plays by the human. At seed.ucsd.edu/ mindreader, a human and computer play this game until one player has won 100 rounds. The "score" is logged as the human's number of rounds won minus the computer's; therefore a negative score means the computer won. In 11,882 games from March 2006 until June 2008, the computer won 86.6% of these games, with an average score of -41.0, which suggests strongly that humans are not able to avoid patterns in their gameplay. Below are a histogram of the games by score as well as by human response time. The second is especially interesting – it suggests that in playing faster, humans more easily fall into predictable patterns.





11 Games as Linear Programs

Note that

$$\min_P \mathbf{M}(P,Q) = \min_i e_i^T \mathbf{M} Q =: b.$$

Therefore the column player seeks to solve the following problem:

$$\begin{aligned} \text{maximize} \quad & \min_i e_i^T \mathbf{M} Q \\ \text{subject to} \quad & \sum_j Q(j) = 1 \\ & Q(j) \geq 0 \qquad j = 1, \ldots, n. \end{aligned}$$

Evidently, this is a linear programming problem. Let's introduce a new variable v, whence the problem can be transformed into

$$\begin{aligned} \text{maximize} \quad & v \\ \text{subject to} \quad & v \leq e_i^T \mathbf{M} Q \\ & \sum_j Q(j) = 1 \\ & Q(j) \geq 0 \qquad j = 1, \ldots, n. \end{aligned}$$

In other words, the column player seeks Q which maximizes the margin of victory.

In matrix notation, if we denote by e the vector of all 1s, we can write this as

$$\begin{aligned} \text{maximize} \quad & v \\ \text{subject to} \quad & ve - \mathbf{M} Q \leq 0 \\ & e^T Q = 1 \\ & Q \geq 0 \qquad j = 1, \ldots, n. \end{aligned}$$

Similarly, the row player seeks $\min_Q(\max_P \mathbf{M}(P,Q))$, which is equivalent to the linear program

$$\begin{aligned} \text{minimize} \quad & u \\ \text{subject to} \quad & ue - \mathbf{M}^T P \geq 0 \\ & e^T P = 1 \\ & P \geq 0 \qquad j = 1, \ldots, n. \end{aligned}$$

Note that u, v can be negative. These problems happen to be *dual* to each other as linear programs. By the minmax theorem, they have the same solution $u^* = v^*$.

In [1], the authors pose a misclassification minimization / margin maximization problem as a linear program and develop a boosting algorithm to solve it in a finite number of steps.

References

[1] Ayhan Demiriz, Kristin P. Bennett, and John Shawe-Taylor. Linear programming boosting via column generation. November 2000.

[2] Robert D. Schapire and Yoav Freund. *Boosting: Foundations and Algorithms*. MIT Press, 2012.