52nd IEEE Conference on Decision and Control
December 10-13, 2013. Florence, Italy

Optimal Adversarial Strategies in Learning with Expert Advice

Anh Truong and Negar Kiyavash

Abstract— **We propose an adversarial setting for the framework of learning with expert advice in which one of the experts has the intention to compromise the recommendation system by providing wrong recommendations. The problem is formulated as a Markov Decision Process (MDP) and solved by dynamic programming. Somewhat surprisingly, we prove that, in the case of logarithmic loss, the optimal strategy for the malicious expert is the greedy policy of lying at every step. Furthermore, a sufficient condition on the loss function is provided that guarantees the optimality of the greedy policy. Our experimental results, however, show that the condition is not necessary since the greedy policy is also optimal when the square loss is used, even though the square loss does not satisfy the condition. Moreover, the experimental results suggest that, for absolute loss, the optimal policy is a threshold one.**

I. INTRODUCTION

Today, online recommendation systems are widely in use. Popular sites such as Netflix and Amazon apply recommendation systems to suggest movies or general merchandise to their clients. At core of these systems, there is a prediction algorithm that, based on recommendations received from a set of experts (users), recommends an item to another user. After the user consumes that item, his feedback is used to assess the performance of experts at that round and refine the predictions of the recommendation system in the future. This general framework of learning from expert advice was introduced by Littlestone and Warmuth [1] and Vovk [2]. Besides online recommendation systems, the framework has been applied to various other applications, for instance, finding the shortest path problem [3], [4], [5] , the metrical task system [6], and online paging [7].

We study an adversarial setup for a two-player recommendation system, where one user tries to sabotage the system which recommends the weighed average of expert opinions. The system initially assigns uniform weights to experts, and updates them over time based on experts' performance and user's feedback. The goal of the malicious expert is to ruin the aggregate quality of the recommendation system over a period of time by providing false recommendations strategically. Note that the greedy approach of providing false recommendation, aka "lie", at each step is not generally optimal as the weight update rule of the recommendation system penalizes experts with poor performance. Thus, one might think that a threshold policy, where the malicious expert tells the "truth" up to some time to gain trust from the system and then lies thereafter, might be optimal.

In this paper, we consider the most basic setup where both outcomes and experts' predictions are binary valued; loss functions are limited to logarithmic, absolute, and square losses. This is because even under this basic setup the problem quickly becomes complex. To find the optimal policy for the malicious expert, we formulate the problem as an MDP and solve it by dynamic programming. We prove that, somewhat surprisingly, when the loss function is logarithmic, the greedy policy of lying is indeed optimal. Moreover, we provide a sufficient condition for the loss function under which the greedy policy is optimal. However, this condition is not necessary as we show through simulations that there exists other loss function, namely, square loss for which the greedy policy is optimal, yet it does not satisfy that condition. We also provide an example of a loss function, the absolute loss, for which the threshold policy is optimal.

The rest of this paper is organized as follows. Section II introduces some related work on learning with expert advice and optimality of greedy policies. Section III describes in detail the notations, algorithms, and formulation of the problem. In Section IV, we prove an important property of value function, which then induces the optimal policy in Section V. Section VI shows our simulation results followed by conclusion and remarks on Section VII.

II. RELATED WORK

Kleinberg et al. considered a stochastic setting for learning with expert advice framework where predictions of experts are drawn from a fixed but unknown distribution [8] . Also in the stochastic setting, Truong et al. proved the convergence of the system to the best expert for a multiplicative weight update rule [9] . Yu et al. introduced a randomized algorithm for a recommender system to diminish the effect of Sybil attackers [10]. However, they make strong assumptions on the fraction of good items and percentage of experts with the same taste as the user to whom the recommendations are made. None of which holds in reality. Study of the adversarial strategies for malicious experts in learning with expert advice framework for the most part is nonexistent. In this paper, we propose such a setting and prove the optimality of a greedy policy for certain loss functions.

Although greedy policies are not usually optimal [11], [12], they are attractive as they are of low complexity and sometimes provide a closed approximation to the optimal solution. Sutton et al [13] proposed an ϵ-greedy algorithm, which has been widely used in multi-arm bandit problems.

A. Truong is with Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, IL 61820, USA truong3@illinois.edu

N. Kiyavash is with Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, IL 61820, USA kiyavash@illinois.edu

Algorithm 1 The weighted average learning algorithm

Input: Set of expert $E = \{1, 2\}$
Initialize: $p_0^i = 1$ for i = 1,2.
for each round $k = 1, 2, ...$ **do**
Nature chooses an item.
Prediction:
Each expert i predicts $x_k^i \in \{0, 1\}$.
Algorithm predicts $\hat{y}_k$,

$$\hat{y}_k = \frac{\sum_{i \in E} p_{k-1}^i x_k^i}{\sum_{i \in E} p_{k-1}^i}. \quad (1)$$

Nature reveals the outcome $y_k \in \{0, 1\}$.
Update:
Algorithm updates weights of all experts. Each weight is updated by

$$p_k^i = \begin{cases} p_{k-1}^i x_k^i & \text{if } y_k = 1, \\ p_{k-1}^i (1 - x_k^i) & \text{if } y_k = 0. \end{cases} \quad (2)$$

end for

Mersereau et al [14] proved that the greedy policy asymptotically coincides with the optimal policy for a class of infinite horizon multi-arm bandit problems. Liu et al [15] analyzed a condition on the discounted factor β for the optimality of greedy policy when a class of so-called standard reward function is used.

III. PROBLEM FORMULATION

Let $E = \{1, 2\}$ be the set of 2 experts. We assume that both experts are always available. At round k, expert i has weight $p_k^i \in [0, 1]$, and his recommendation on a given item is denoted by $x_k^i \in \{0, 1\}$. Denote $\vec{p}_k$ as the weight vector at k, i.e., $\vec{p}_k = (p_k^1, p_k^2)$. Define $\tilde{p}_k^1 = \frac{p_k^1}{p_k^1 + p_k^2}$, i.e., $\tilde{p}_k^1$ is the relative weight of expert 1 at step k. Similarly, $\tilde{p}_k^2 = \frac{p_k^2}{p_k^1 + p_k^2}$, and $\vec{\tilde{p}}_k$ is the relative weight vector at k, i.e., $\vec{\tilde{p}}_k = (\tilde{p}_k^1, \tilde{p}_k^2)$. The true outcome, or user's feedback on a given item, is denoted by y_k, which is also considered to value in $\{0, 1\}$.

The learning process is given in Algorithm 1. The system uses a weighted average rule for predictions by (1) and a multiplicative rule for weights update by (2). After the true outcome for the item is revealed, the system incurs a loss $l(\hat{y}_k, y_k)$. One example of such a loss function is the logarithmic loss defined as,

$$l(\hat{y}_k, y_k) = -\mathbb{I}\{y_k = 1\} \ln \hat{y}_k - \mathbb{I}\{y_k = 0\} \ln(1 - \hat{y}_k). \quad (3)$$

In the case of logarithmic loss, to avoid the loss function going to infinity, we slightly modify the binary predictions to $\{\epsilon, 1 - \epsilon\}$, where ϵ is a small number. We assume that expert 2 makes a correct recommendation, i.e., one that agrees with users feedback, with probability μ_2, the accuracy of expert 2, or

$$x_k^2 = \begin{cases} y_k & \text{w.p } \mu_2, \\ 1 - y_k & \text{w.p } 1 - \mu_2. \end{cases} \quad (4)$$

As we are interested in the performance of the system for an adversarial setup, we assume expert 1 knows the true outcome, y_k, at time k, as well as the predictions distribution of expert 2 (this can be learned empirically from the history of predictions, for example.) This is a stronger adversarial model than models of which a malicious has to mislead the system without knowledge of the outcome. Moreover, the whole history of predictions before time k is also available to expert 1, i.e., he knows $\vec{p}_{l-1}$, for $l = 1, ..., k-1$ and consequently $\vec{\tilde{p}}_{l-1}^1$, for $l = 1, ..., k-1$. In summary, at time k, the pair of variables $(y_k, \tilde{p}_{k-1}^1) := s(k)$ is known to expert 1. Note that, since $\tilde{p}_{k-1}^1 + \tilde{p}_{k-1}^2 = 1$, knowing $\tilde{p}_{k-1}^1$ is sufficient for expert 1 to calculate the relative weight vector $\vec{\tilde{p}}_{k-1}$. This expert then seeks an optimal policy to maximize the accumulated loss of the system up to horizon N by choosing a sequence of predictions $\{x_k^1\}_1^N$. The problem is formulated as a MDP, in which, at each time k, expert 1 selects an action (prediction) $x_k^1 = f(s(k)) \in \{T, L\}$, where

$$\begin{cases} T = \mathbb{I}\{y_k = 1\}(1 - \epsilon) + \mathbb{I}\{y_k = 0\}\epsilon, \\ L = \mathbb{I}\{y_k = 1\}\epsilon + \mathbb{I}\{y_k = 0\}(1 - \epsilon). \end{cases} \quad (5)$$

The objective function of expert 1 then becomes

$$\max_{x_k^1(s(k)), k=1...N} \sum_{k=1}^{N} \mathbb{E}_{x_k^2}(l(\hat{y}_k, y_k)), \quad (6)$$

where the expectation is taken over x_k^2. The state transition associated with this MDP is given by $s(k+1) = \phi(s(k), x_k^1)$. Note that transition of $\{y_k\}$ is driven by an arbitrary function known to expert 1. From (2), the evolution of relative weight $\tilde{p}_{k-1}^1$ is given by:

$$\tilde{p}_k^1 = \begin{cases} \frac{p_{k-1}^1 x_k^1}{p_{k-1}^1 x_k^1 + p_{k-1}^2 x_k^2} & \text{if } y_k = 1, \\ \frac{p_{k-1}^1 (1 - x_k^1)}{p_{k-1}^1 (1 - x_k^1) + p_{k-1}^2 (1 - x_k^2)} & \text{if } y_k = 0. \end{cases} \quad (7)$$

It is clear from (7) that, when two experts make the same prediction at one time, their next (updated) relative weights do not change. Otherwise, expert 1's relative weight increases if he makes the right recommendation while expert 2 makes a wrong one, or his relative weight decreases if the converse is true. Specifically,

$$\tilde{p}_k^1 = \begin{cases} \frac{1}{1 + \left(\frac{1}{\tilde{p}_{k-1}^1} - 1\right) \frac{1-\epsilon}{\epsilon}} & \text{if } x_k^1 = L, x_k^2 = T, \\ \frac{1}{1 + \left(\frac{1}{\tilde{p}_{k-1}^1} - 1\right) \frac{\epsilon}{1-\epsilon}} & \text{if } x_k^1 = T, x_k^2 = L, \\ \tilde{p}_{k-1}^1 & \text{if } x_k^1 = x_k^2. \end{cases} \quad (8)$$

The dynamic program which solves MDP problem of (6) is given in Algorithm 2. In Algorithm 2, $c_k(y_k, \tilde{p}_{k-1}^1, x_k^1)$ denotes the current cost and is defined as

$$c_k(y_k, \tilde{p}_{k-1}^1, x_k^1) = \mathbb{E}_{x_k^2}(l(\hat{y}_k, y_k)). \quad (11)$$

$V_{k+1}^*(\phi(y_k, \tilde{p}_{k-1}^1, x_k^1))$ denotes the value function, the optimally accumulated loss from $k + 1$ to the final time N. Define $V_k(y_k, \tilde{p}_{k-1}^1, x_k^1)$ by

$$V_k(y_k, \tilde{p}_{k-1}^1, x_k^1) = c_k(y_k, \tilde{p}_{k-1}^1, x_k^1) + \mathbb{E}_{x_k^2} V_{k+1}^*(\phi(y_k, \tilde{p}_{k-1}^1, x_k^1)). \quad (12)$$

Algorithm 2 Adversary's optimal strategy (dynamic program)

Initialize: $V_N(.) = c_N(.) = 0$.
for each step $k = N-1$ downto 1 **do**
Find the optimal action,

$$u_k^*(y_k, \tilde{p}_{k-1}^1) = \arg\max_{x_k^1} \left[c_k(y_k, \tilde{p}_{k-1}^1, x_k^1) + \mathbb{E}_{x_k^2} V_{k+1}^*(\phi(y_k, \tilde{p}_{k-1}^1, x_k^1))\right], \quad (9)$$

and the corresponding value function,

$$V_k^*(y_k, \tilde{p}_{k-1}^1) = \max_{x_k^1} \left[c_k(y_k, \tilde{p}_{k-1}^1, x_k^1) + \mathbb{E}_{x_k^2} V_{k+1}^*(\phi(y_k, \tilde{p}_{k-1}^1, x_k^1))\right]. \quad (10)$$

Output: sequence $u_{N-1}^*(.), V_{N-1}^*(.), ..., u_0^*(.), V_0^*(.)$.
end for

From (3), (1) and (4), the current cost in (11) is derived as $c_k(y_k, \tilde{p}_{k-1}^1, x_k^1)$

$$\begin{aligned} &= -\mathbb{I}\{y_k = 1\} \left[\mu_2 \ln \frac{p_{k-1}^1 x_k^1 + p_{k-1}^2(1-\epsilon)}{p_{k-1}^1 + p_{k-1}^2} \right. \\ &\qquad \left. + (1-\mu_2) \ln \frac{p_{k-1}^1 x_k^1 + p_{k-1}^2 \epsilon}{p_{k-1}^1 + p_{k-1}^2}\right] \\ &\quad - \mathbb{I}\{y_k = 0\} \left[\mu_2 \ln \frac{p_{k-1}^1 (1-x_k^1) + p_{k-1}^2(1-\epsilon)}{p_{k-1}^1 + p_{k-1}^2} \right. \\ &\qquad \left. + (1-\mu_2) \ln \frac{p_{k-1}^1 (1-x_k^1) + p_{k-1}^2 \epsilon}{p_{k-1}^1 + p_{k-1}^2}\right]. \quad (13) \end{aligned}$$

To simplify the notation, let us define $c_L(\tilde{p}_{k-1}^1) := c_k(y_k, \tilde{p}_{k-1}^1, L)$ and $c_T(\tilde{p}_{k-1}^1) := c_k(y_k, \tilde{p}_{k-1}^1, T)$. From (13) and (5), the current costs can be written as

$$c_L(\tilde{p}_{k-1}^1) = -\mu_2 \ln\left(\tilde{p}_{k-1}^1(2\epsilon - 1) + 1 - \epsilon\right) - (1-\mu_2)\ln\epsilon, \quad (14)$$

and

$$c_T(\tilde{p}_{k-1}^1) = -\mu_2 \ln(1-\epsilon) - (1-\mu_2)\ln\left(\tilde{p}_{k-1}^1(1-2\epsilon) + \epsilon\right). \quad (15)$$

One can easily verify the following two properties of the current costs,

- **(P1)**: $c_L(\tilde{p}_{k-1}^1)$ is an increasing function of $\tilde{p}_{k-1}^1$.
- **(P2)**: $c_L(\tilde{p}_{k-1}^1) \geq c_T(\tilde{p}_{k-1}^1)$ for all $\tilde{p}_{k-1}^1$.

The terminal cost is set to be zero. Since finding closed-form value functions proves intractable, we attempt to guess the structures of optimal actions as well as value function using policy iteration.

IV. EFFECTS OF RELATIVE WEIGHTS ON VALUE FUNCTION

In this section, we see how the change in relative weights influences the value function, and hence, the optimal policy.

Theorem 1. *Given $\vec{p}_{k-1}$ and $\vec{q}_{k-1}$ such that $\tilde{q}_{k-1}^1 \leq \tilde{p}_{k-1}^1$, and when a logarithmic loss function is used,*

$$V_k^*(y_k, \tilde{q}_{k-1}^1) \leq V_k^*(y_k, \tilde{p}_{k-1}^1).$$

Proof. The theorem is proved by induction.

- The base case: at the final step $N-1$, by (10), the value function is given as

$$\begin{aligned} V_{N-1}^*(y_{N-1}, \tilde{p}_{N-2}^1) &= \max_{x_{N-1}^1} \left[c_{N-1}(y_{N-1}, \tilde{p}_{N-2}^1, x_{N-1}^1) \right. \\ &\qquad \left. + \mathbb{E}_{x_{N-1}^2} V_N^*(\phi(y_{N-1}, \tilde{p}_{N-2}^1, x_{N-1}^1))\right], \\ &= \max_{x_{N-1}^1} c_{N-1}(y_{N-1}, \tilde{p}_{N-2}^1, x_{N-1}^1), \quad (16) \end{aligned}$$

since the terminal cost is assumed to be zero. Note that from evaluating (13) at $k = N-1$, we see that $c_{N-1}(y_{N-1}, \tilde{p}_{N-2}^1, x_{N-1}^1)$ is a decreasing function of x_{N-1}^1 when $y_{N-1} = 1$, and an increasing function of x_{N-1}^1 when $y_{N-1} = 0$. Thus, the optimal decision at step $N-1$ is to tell a lie,

$$u_{N-1}^*(y_{N-1}, \tilde{p}_{N-2}^1) = \begin{cases} \epsilon & \text{if } y_{N-1} = 1, \\ 1-\epsilon & \text{if } y_{N-1} = 0. \end{cases} \quad (17)$$

Then, combining (16) and (14), we have
$V_{N-1}^*(y_{N-1}, \tilde{p}_{N-2}^1)$

$$= -\mu_2 \ln\left(\tilde{p}_{N-2}^1(2\epsilon - 1) + 1 - \epsilon\right) - (1-\mu_2)\ln\epsilon. \quad (18)$$

Since $V_{N-1}^*(.)$ is an increasing function of $\tilde{p}_{N-2}^1$, the base case follows.

- Now, assume theorem 1 holds at step $k+1$, i.e., if $\tilde{q}_k^1 \leq \tilde{p}_k^1$, then $V_{k+1}^*(y_{k+1}, \tilde{q}_k^1) \leq V_{k+1}^*(y_{k+1}, \tilde{p}_k^1)$, we show that it also holds at step k.
From property **(P1)** of the current costs,

$$c_L(\tilde{q}_{k-1}^1) \leq c_L(\tilde{p}_{k-1}^1). \quad (19)$$

Furthermore, if at step k, expert 1 tells a lie at both states $\tilde{p}_{k-1}^1$ and $\tilde{q}_{k-1}^1$, the updated relative weights satisfy $\tilde{q}_k^1 \leq \tilde{p}_k^1$ by (8). It follows from the induction assumption that

$$\mathbb{E}V_{k+1}^*(\phi(y_k, \tilde{q}_{k-1}^1, L)) \leq \mathbb{E}V_{k+1}^*(\phi(y_k, \tilde{p}_{k-1}^1, L)). \quad (20)$$

Inequalities (19) and (20) imply that

$$V_k(y_k, \tilde{q}_{k-1}^1, L) \leq V_k(y_k, \tilde{p}_{k-1}^1, L), \quad (21)$$

or

$$V_k(y_k, \tilde{q}_{k-1}^1, L) \leq \max\{V_k(y_k, \tilde{p}_{k-1}^1, L), V_k(y_k, \tilde{p}_{k-1}^1, T)\}.$$

The proof will be completed if we also prove that

$$V_k(y_k, \tilde{q}_{k-1}^1, T) \leq \max\{V_k(y_k, \tilde{p}_{k-1}^1, L), V_k(y_k, \tilde{p}_{k-1}^1, T)\}.$$

We do this by contradiction. Suppose that,

$$V_k(y_k, \tilde{q}_{k-1}^1, T) > V_k(y_k, \tilde{p}_{k-1}^1, T), \quad (22)$$

and

$$V_k(y_k, \tilde{q}_{k-1}^1, T) > V_k(y_k, \tilde{p}_{k-1}^1, L). \quad (23)$$

From (22) and (12), we observe that,

$$\begin{aligned} &c_T(\tilde{q}_{k-1}^1) - c_T(\tilde{p}_{k-1}^1) > \\ &\mathbb{E}V_{k+1}^*(\phi(y_k, \tilde{p}_{k-1}^1, T)) - \mathbb{E}V_{k+1}^*(\phi(y_k, \tilde{q}_{k-1}^1, T)). \quad (24) \end{aligned}$$

On the other hand, from (23) and (21),

$$V_k(y_k, \tilde{q}^1_{k-1}, T) > V_k(y_k, \tilde{q}^1_{k-1}, L).$$

It follows that

$$\begin{aligned}
&\mathbb{E}V^*_{k+1}(\phi(y_k, \tilde{q}^1_{k-1}, T)) - \mathbb{E}V^*_{k+1}(\phi(y_k, \tilde{q}^1_{k-1}, L)) \\
&\overset{(a)}{>} c_L(\tilde{q}^1_{k-1}) - c_T(\tilde{q}^1_{k-1}), \\
&\overset{(b)}{>} c_L(\tilde{u}^1_{k-1}) - c_T(\tilde{q}^1_{k-1}), \\
&\overset{(c)}{>} c_T(\tilde{u}^1_{k-1}) - c_T(\tilde{q}^1_{k-1}), \\
&\overset{(d)}{>} \mathbb{E}V^*_{k+1}(\phi(y_k, \tilde{q}^1_{k-1}, T)) - \mathbb{E}V^*_{k+1}(\phi(y_k, \tilde{u}^1_{k-1}, T)), \quad (25)
\end{aligned}$$

where $\vec{u}_{k-1}$ is a specific state at k such that $u^1_{k-1} := \frac{\epsilon}{1-\epsilon} q^1_{k-1} < q^1_{k-1}$, which implies $\tilde{u}^1_{k-1} < \tilde{q}^1_{k-1}$. Inequalities (a), (b), (c) follow from (12), properties **(P1)**, and **(P2)** respectively. (d) follows from the argument of (24). Recall from the update rule (2), when every times expert 1 tells the truth, his weight will be updated by the factor $1-\epsilon$. Hence,

$$u^1_k = u^1_{k-1}(1-\epsilon) = q^1_{k-1}\epsilon = q^1_k.$$

It follows that

$$\mathbb{E}V^*_{k+1}(\phi(y_k, \tilde{q}^1_{k-1}, L)) = \mathbb{E}V^*_{k+1}(\phi(y_k, \tilde{u}^1_{k-1}, T)),$$

and thus implies a contradiction on (25). This concludes the proof.

□

V. OPTIMAL POLICY FOR LOGARITHMIC LOSS

A greedy policy is defined as follows.

Definition 1. *Greedy policy is a policy in which expert* 1 *makes his decisions based only on the current costs.*

Property **(P2)** of current costs and Theorem 1 implies the trade-off between current costs and value function. Specifically, expert 1 can cause the system to incur a higher current cost if he tells a lie rather than the truth, but then his updated weight decreases and as does his value function (by Theorem 1) as a consequence. Therefore, it might be optimal to gain system trust up to a certain value before starting to lie. Interestingly, for the logarithmic loss, we prove that the greedy policy of always lying is optimal.

Let p_m denote a realization of $\tilde{p}^1_{k-1}$, then from (8), we have

$$\begin{aligned}
p_{m+1} &= 1 \Big/ \left(1 + \left(\frac{1}{p_m} - 1\right) \frac{\epsilon}{1-\epsilon}\right) \text{ if } x^1_k = T, x^2_k = L, \\
&= p_m(1-\epsilon) / \left(p_m(1-2\epsilon) + \epsilon\right), \quad (26)
\end{aligned}$$

and

$$\begin{aligned}
p_{m-1} &= 1 \Big/ \left(1 + \left(\frac{1}{p_m} - 1\right) \frac{1-\epsilon}{\epsilon}\right) \text{ if } x^1_k = L, x^2_k = T, \\
&= p_m\epsilon / \left(p_m(2\epsilon - 1) + 1 - \epsilon\right). \quad (27)
\end{aligned}$$

We will use these notations in our proof. Note that from (26) and (27), $p_{m-1} < p_m < p_{m+1}$. Let $\mathcal{A} = \{..., p_{m-1}, p_m, p_{m+1}, ...\}$ denote the state space of relative weights of expert 1, where the one-step state transition follows (26) and (27). The probability transition matrix is given by

$$\begin{cases}
P_{m,m-1}(u) = \mu_2 \mathbb{I}\{u = L\}, \\
P_{m,m}(u) = \mu_2 \mathbb{I}\{u = T\} + (1-\mu_2)\mathbb{I}\{u = L\}, \\
P_{m,m+1}(u) = (1-\mu_2)\mathbb{I}\{u = T\}.
\end{cases} \quad (28)$$

where u is the prediction of expert 1. Rewrite current costs in (14) and (15), using this notation, as

$$\begin{cases}
c_L(p_m) = -\mu_2 \ln\left(p_m(2\epsilon - 1) + 1 - \epsilon\right) - (1-\mu_2)\ln\epsilon, \\
c_T(p_m) = -\mu_2 \ln(1-\epsilon) - (1-\mu_2)\ln\left(p_m(1-2\epsilon) + \epsilon\right).
\end{cases}$$

Theorem 2. *For a logarithmic loss applied in Algorithm 1 and Algorithm 2, the adversary's optimal policy is a greedy policy, one which is optimal to tell a lie at every step.*

Proof. It suffices to prove

$$V_k(y_k, p_m, L) > V_k(y_k, p_m, T) \text{ for all } k \text{ and } p_m, \quad (29)$$

or equivalently by (12) to prove

$$\Delta V^*_{k+1}(y_k, p_m) < \Delta c(p_m), \quad (30)$$

where $\Delta c(p_m) = c_L(p_m) - c_T(p_m)$, and
$\Delta V^*_{k+1}(y_{k+1}, p_m) = \mathbb{E}V^*_{k+1}(\phi(y_k, p_m, T)) - \mathbb{E}V^*_{k+1}(\phi(y_k, p_m, L))$.
We note that from (28),

$$\begin{aligned}
&\Delta V^*_{k+1}(y_{k+1}, p_m) \\
&= \mu_2 \Big(V^*_{k+1}(y_{k+1}, p_m) - V^*_{k+1}(y_{k+1}, p_{m-1}) \Big), \\
&+ (1-\mu_2) \Big(V^*_{k+1}(y_{k+1}, p_{m+1}) - V^*_{k+1}(y_{k+1}, p_m) \Big). \quad (31)
\end{aligned}$$

We prove (30) by induction.

- Base case: at step $N-1$. Due to the zero assumption of the terminal cost and property **(P1)** of current costs, the claim holds true.
- Induction step:
  Assume $\Delta V^*_{k+2}(y_{k+2}, p_m) < \Delta c(p_m)$. We need to prove that (30) is true for step k. Since at step $k+1$, the optimal decision is to lie at every state, from (31), we have

$$\begin{aligned}
&\Delta V^*_{k+1}(y_{k+1}, p_m) \\
&= \mu_2 \big[c_L(p_m) \\
&\quad + \mu_2 V^*_{k+2}(y_{k+2}, p_{m-1}) + (1-\mu_2) V^*_{k+2}(y_{k+2}, p_m)\big] \\
&\quad - \mu_2 \big[c_L(p_{m-1}) \\
&\quad + \mu_2 V^*_{k+2}(y_{k+2}, p_{m-2}) + (1-\mu_2) V^*_{k+2}(y_{k+2}, p_{m-1})\big] \\
&\quad + (1-\mu_2)\big[c_L(p_{m+1}) \\
&\quad + \mu_2 V^*_{k+2}(y_{k+2}, p_m) + (1-\mu_2) V^*_{k+2}(y_{k+2}, p_{m+1})\big] \\
&\quad - (1-\mu_2)\big[c_L(p_m) \\
&\quad + \mu_2 V^*_{k+2}(y_{k+2}, p_{m-1}) + (1-\mu_2) V^*_{k+2}(y_{k+2}, p_m)\big] \\
&= \mu_2 \left[c_L(p_m) - c_L(p_{m-1})\right] + \mu_2 \Delta V^*_{k+2}(y_{k+2}, p_{m-1}) \\
&\quad + (1-\mu_2)\left[c_L(p_{m+1}) - c_L(p_m)\right] \\
&\qquad + (1-\mu_2)\Delta V^*_{k+2}(y_{k+2}, p_m)
\end{aligned}$$

$$\begin{aligned}
&\overset{(a)}{<} \mu_2\left[c_L(p_m) - c_L(p_{m-1})\right] + \mu_2\Delta c(p_{m-1}) \\
&\quad + (1-\mu_2)\left[c_L(p_{m+1}) - c_L(p_m)\right] + (1-\mu_2)\Delta c(p_m) \\
&= \mu_2\left[c_L(p_m) - c_T(p_{m-1})\right] \\
&\qquad + (1-\mu_2)\left[c_L(p_{m+1}) - c_T(p_m)\right], \quad (32)
\end{aligned}$$

where (a) follows from the induction assumptions $\Delta V^*_{k+2}(y_{k+2}, p_{m-1}) < \Delta c(p_{m-1})$ and $\Delta V^*_{k+2}(y_{k+2}, p_m) < \Delta c(p_m)$. The proof will be completed if we can show that the RHS of (32) is equal to $\Delta c(p_m)$.
Indeed,

$$\begin{aligned}
&\mu_2\left[c_L(p_m) - c_T(p_{m-1})\right] + (1-\mu_2)\left[c_L(p_{m+1}) - c_T(p_m)\right] \\
&\qquad = c_L(p_m) - c_T(p_m) \\
&\Leftrightarrow (1-\mu_2)\Big[c_L(p_{m+1}) - c_L(p_m)\Big] = \mu_2\Big[c_T(p_{m-1}) - c_T(p_m)\Big] \\
&\Leftrightarrow \ln\frac{p_m(2\epsilon-1)+1-\epsilon}{p_{m+1}(2\epsilon-1)+1-\epsilon} = \ln\frac{p_m(1-2\epsilon)+\epsilon}{p_{m-1}(1-2\epsilon)+\epsilon} \\
&\Leftrightarrow p_m(1-\epsilon)(2\epsilon-1) + (1-\epsilon)(1-2\epsilon)p_m + (1-\epsilon)\epsilon \\
&\qquad = p_m\epsilon(1-2\epsilon) + \epsilon(1-\epsilon) + \epsilon(2\epsilon-1)p_m.
\end{aligned}$$

Since the last equality is true, the proof is completed. □

Note that optimality result of the Theorem 2 extends to any general loss as long as it satisfies the following condition:

$$(1-\mu_2)\Big[c_L(p_{m+1}) - c_L(p_m)\Big] \le \mu_2\Big[c_T(p_{m-1}) - c_T(p_m)\Big]. \quad (33)$$

While (33) is a sufficient condition, it is not a necessary one. In the next section, we will show through our experimental results that at least for one other kind of loss, namely square loss, the greedy policy is optimal but the square loss does not satisfy (33).

VI. EXPERIMENTAL RESULTS

In this section, we introduce the experimental results with three kinds of losses: logarithmic loss, square loss, and absolute loss. A synthetic dataset including 100 recommendation steps was generated. The accuracy μ_2 of expert 2 is varied from 0 to 1. For each type of loss, we run the dynamic program of Algorithm 2 and find the optimal policy by comparing the difference of current costs and the difference of value functions at every state p_m.

A. Logarithmic loss

Recall that to avoid the loss going to infinity, we used predictions $\{\epsilon, 1-\epsilon\}$ for experts. Here, we fix $\epsilon = 0.01$. Figure 1 shows the optimal policy for this loss when $\mu_2 = 0.5$. In Figure 1, the dash curve represents the difference of current costs while the solid curves are the difference of value functions at different steps. It can be seen that telling a lie is optimal since the difference of current costs always exceeds the difference of value functions corresponding to lying and telling the truth for every step.

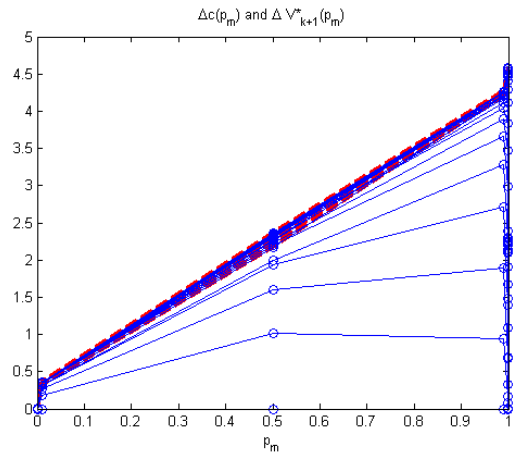

Fig. 1. Difference of current costs v.s difference of value functions for logarithmic loss ($\mu_2 = 0.5$).

B. Square loss

Loss of the algorithm at step k is defined as $l(\hat{y}_k) = (\hat{y}_k - y_k)^2$.
The weight update rule (2) becomes

$$p^i_k = \begin{cases} p^i_{k-1}e^{-1} & \text{if } x^1_k = L, \\ p^i_{k-1} & \text{if } x^1_k = T. \end{cases}$$

Figure 2 shows the experimental results for the square loss when $\mu_2 = 0.5$. We can see from Figure 2 that, the square

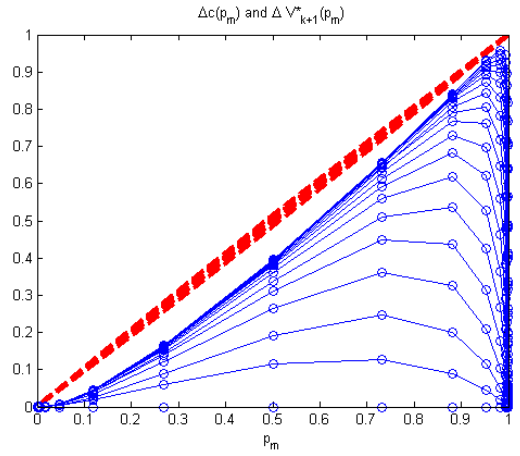

Fig. 2. Difference of current costs v.s difference of value functions for square loss ($\mu_2 = 0.5$).

loss has the same optimal policy as the logarithmic loss, i.e., telling a lie is always optimal. Since this square loss does not satisfy (33), it also points out that (33) is not a necessary condition.

C. Absolute loss

Loss of the algorithm at step k is defined as $l(\hat{y}_k) = |\hat{y}_k - y_k|$. The weight update rule (2) becomes

$$p^i_k = \begin{cases} p^i_{k-1}e^{-1} & \text{if } x^1_k = L, \\ p^i_{k-1} & \text{if } x^1_k = T. \end{cases}$$

For this loss, the optimal policy is a threshold policy, where expert 1 tells the truth if his relative weight is less than a certain value, and tells a lie otherwise. Experimental results are depicted in Figures 3 and 4, where $\mu_2 = 0.2$ and 0.5, respectively. In Figure 3 and 4, there exists a threshold

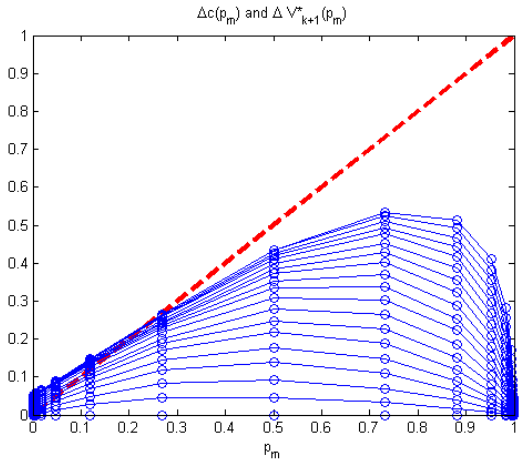

Fig. 3. Difference of current costs v.s difference of value functions for absolute loss ($\mu_2 = 0.2$).

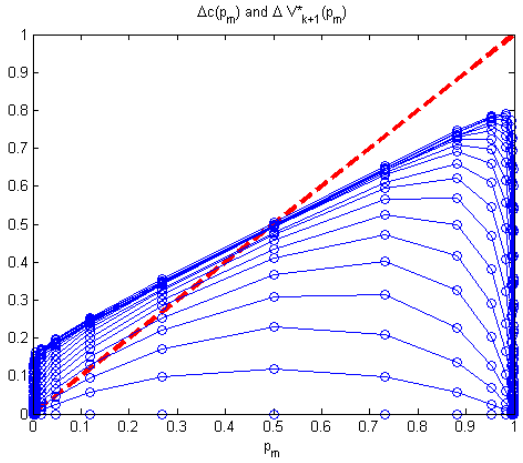

Fig. 4. Difference of current costs v.s difference of value functions for absolute loss ($\mu_2 = 0.5$).

point, where for states below this point, the difference of value functions is greater than the difference of current costs, and thus the optimal action of expert 1 is to tell the truth. Conversely, at states above this point, expert 1 must tell a lie. Observe that the threshold is non-decreasing in μ_2 (due to the limited space, some other figures corresponding to different values of μ_2 are omitted). It follows from the fact that when expert 2 is more often accurate, the malicious expert (expert 1) should gain more trust before starting to lie to be able to dominate the system's recommendation.

VII. Conclusions and Remarks

We consider a recommendation system with two experts, one of them honest and the other malicious. The malicious expert tries to sabotage the system by giving wrong predictions. We derives the optimal policy for the malicious expert, i.e., one which maximizes the total loss of the system. The problem is formulated as an MDP, and solved by dynamic programming. Three kinds of losses are investigated in this paper. With the logarithmic loss, interestingly, we prove that the optimal strategy is the greedy policy of telling a lie at all times. Further, we establish a sufficient condition for optimality of this greedy policy for a general loss function. Through simulation results, we show that this condition is not necessary since the greedy policy is optimal for the square loss, even though it does not satisfy the condition. Our experimental results also show that for the absolute loss, the optimal policy is a threshold policy.

VIII. Acknowledgments

This work was funded in part by AFOSR grants FA 9550-11-1-0016, FA 9550-10-1-0573, and FA 9550- 10-1-0345,; and by NSF grant CCF 10-54937 CAR.

References

[1] N. Littlestone and M. K. Warmuth, "The weighted majority algorithm," in *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, NC, Feb. 1989, pp. 212–261.

[2] V. G. Vovk, "Aggregating strategies," in *Proceedings of the third annual workshop on Computational learning theory (COLT '90)*, San Francisco, CA, USA, 1990, pp. 371–386.

[3] A. György and G. Ottucsák, "Adaptive routing using expert advice," *Computer Journal*, vol. 49, pp. 180–189, Mar. 2006.

[4] B. Awerbuch and R. Kleinberg, "Adaptive routing with end-to-end feedback: distributed learning and geometric approaches," in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, New York, NY, USA, 2004, pp. 45–53.

[5] A. Kalai and S. Vempala, "Efficient algorithms for online decision problems," *Journal of Computer and System Sciences*, vol. 71, pp. 291–307, Oct. 2005.

[6] A. Blum and C. Burch, "On-line learning and the metrical task system problem," in *Proceedings of the tenth annual conference on Computational learning theory*, New York, NY, USA, 1997, pp. 45–53.

[7] A. Blum, C. Burch, and A. Kalai, "Finely-competitive paging," in *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, Washington, DC, USA, 1999, pp. 450–.

[8] R. D. Kleinberg, A. Niculescu-Mizil, and Y. Sharma, "Regret bounds for sleeping experts and bandits," in *Proceedings of the 21st Annual Conference on Learning Theory - COLT 2008*, Helsinki, Finland, 2008, pp. 425–436.

[9] A. Truong, N. Kiyavash, and V. Borkar, "Convergence analysis for an online recommendation system," in *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference*, Orlando, Florida, US, Dec. 2011, pp. 3889 – 3894.

[10] H. Yu, C. Shi, M. Kaminsky, P. B. Gibbons, and F. Xiao, "Dsybil: Optimal sybil-resistance for recommendation systems," in *Proceedings of the 2009 30th IEEE Symposium on Security and Privacy*, Washington, DC, USA, 2009, pp. 283–298.

[11] J. Bang-Jensen, G. Gutin, and A. Yeo, "When the greedy algorithm fails," *Discrete Optimization*, vol. 1, pp. 121 – 127, 2004.

[12] G. Gutin, A. Yeo, and A. Zverovich, "Traveling salesman should not be greedy: domination analysis of greedy-type heuristics for the tsp," *Discrete Applied Mathematics*, vol. 117, pp. 81–86, 2002.

[13] R. S. Sutton and A. G. Barto, *Reinforcement Learning I: Introduction*. The MIT Press, 1998.

[14] A. J. Mersereau, P. Rusmevichientong, and J. N. Tsitsiklis, "A structured multiarmed bandit problem and the greedy policy," *IEEE Transactions on Automatic Control*, vol. 54, pp. 2787–2802, 2009.

[15] Q. Liu, K. Wang, and L. Chen, "On optimality of greedy policy for a class of standard reward function of restless multi-armed bandit problem," *CoRR*, vol. abs/1104.5391, 2011.