Advanced Course in Machine Learning Spring 2010

Active Learning: Bandits

Handouts are jointly prepared by Shie Mannor and Shai Shalev-Shwartz

In supervised learning, the goal of the learning algorithm is to learn a predictor, $h : \mathcal{X} \to \mathcal{Y}$, which accurately predicts labels of future instances. In the traditional PAC learning model, the learner receives a training set of examples which are sampled i.i.d. from an unknown distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. The learner has no control on which examples he receives.

In *active* learning the learner interactively chooses and affects the information he obtains from the data source. There are many variants of how the learner can control the information. For example, the learner can ask to label any instance in the instance space, or any specific instance in a given large unlabeled set of examples.

The main motivation of using active learning is because in many practical problems, obtaining unlabeled examples is cheap while labeling an instance is expensive (e.g. because it requires manual labeling by humans). For example, consider the problem of object recognition in vision. Nowadays, we can obtain an enormous number of unlabeled images by simply placing a camera, but labeling these images might be very expensive.

We first present simple examples in which active learning can provably help reducing the sample complexity (or, here, it is in fact the "label complexity"). Next, we will describe a related problem, which is called "the multi-armed bandit problem", of independent interest, and will discuss its relatedness to active learning. Finally, we will describe and analyze several active learning algorithms.

1 Active learning can help

In this section we show that in some cases active learning can significantly reduce the number of labels required for learning.

Consider the problem of learning the class of thresholds on the real line, $\mathcal{H} = \{x \mapsto \text{sign}(x - w) : w \in \mathbb{R}\}$. As we have learned in the previous course, the VC dimension of $\mathcal{H}$ is 1 and therefore the sample complexity of (passive) PAC learning $\mathcal{H}$ is $\Theta\left(\frac{\log(1/\delta)}{\epsilon}\right)$.

Now, lets describe an active learning algorithm for this problem. We will start with sampling $m_u = \Theta\left(\frac{\log(1/\delta)}{\epsilon}\right)$ *unlabeled* points. Then, we will find a hypothesis that has zero training error using a binary search.

Note that the number of labels we will ask is $O(\log(m_u))$. Additionally, since we found an ERM hypothesis we have the same PAC learning guarantees. Therefore, we reduced the label complexity from $\Theta\left(\frac{\log(1/\delta)}{\epsilon}\right)$ to $\Theta\left(\log\left(\frac{\log(1/\delta)}{\epsilon}\right)\right)$. This is an exponential improvement!

The above example shows how by efficiently searching through hypothesis class, active learning can reduce the label complexity. Another idea is to exploit structure in the data. For example, one can first train a clustering algorithm over the unlabeled data set. Then, if we are luck and the data is well clustered, we can ask for the labels of a small number of instances in each class.

2 The Stochastic Multi-Armed Bandit Problem

The multi-armed bandit problem is defined as follows. We have d gambling machines (i.e. the "arm" of a bandit) , each of which yields a random reward with distribution over $[0, 1]$ and an expectation of μ_i. The gambler can pull totally n arms and his goal is to maximize his total reward. Formally, on each round the gambler chooses one of the arms, pulls it, and receives a reward which is an instantiation of the random variable associated with the chosen arm.

A policy A is an algorithm that chooses the next arm to pull based on the sequence of past plays and rewards. Let $(a_1, r_1), \ldots, (a_n, r_n)$ denote the sequence of arm-reward pairs after running the policy for n rounds. The total reward of the algorithm is $\sum_{t=1}^{n} r_t$. Note that the total reward is a random variable. A good policy is one that has large expected reward. Of course, the best policy is to always pull the arm with maximal expected reward. That is, if $\mu^\star = \max_j \mu_j$ and $j^\star$ is an arm that achieves the maximum, then the best policy always pulls $j^\star$ and achieves an expected reward of $n\mu^\star$. We define the regret of a policy, for not choosing always the best arm, to be

$$n\mu^\star - \mathbb{E}\left[\sum_{i=1}^{n} r_t\right] .$$

We now describe several strategies for the multi-armed problem.

2.1 First learn then test

Our first approach is straightforward. We start with a learning phase in which we pull each arm exactly m rounds and estimate μ_i using the obtained data according to the ERM rule, namely,

$$\hat{\mu}_i \;=\; \frac{1}{m} \sum_{t=(i-1)m+1}^{im} r_t \; .$$

After the learning step, we shall simply pull the arm with the largest estimated reward for the rest of the $n-m$ rounds. That is, we will pull some $\hat{i} \in \operatorname{argmax}_i \hat{\mu}_i$.

Lets analyze the regret of this approach. Using Hoeffding's concentration bound we know that for each individual arm:

$$\mathbb{P}[|\hat{\mu}_i - \mu_i| > \epsilon] \le 2e^{-m\epsilon^2} \; .$$

Applying the union bound, we get that

$$\mathbb{P}[\exists i,\ |\hat{\mu}_i - \mu_i| > \epsilon] \le 2de^{-m\epsilon^2} \; .$$

This implies that with probability of at least $1 - 2de^{-m\epsilon^2}$, we will have that $\mu_{\hat{i}} \ge \mu^\star - 2\epsilon$. In particular, if $\gamma = \mu^\star - \max_{j \neq j^\star} \mu_j$, then for $\epsilon \le \gamma/2$ we will choose the optimal arm with probability of at least $1 - 2de^{-m\epsilon^2}$.

Denote $\delta = 2de^{-m\epsilon^2}$, then with probability of at least $1 - \delta$ we have that the average regret of the algorithm over the last $n - m$ rounds is at most $2\epsilon(n - m)$. And, with probability of at most δ the regret of the algorithm over the last $n - m$ rounds is at most $(n - m)\mu^\star$. Additionally, it is clear that on the first m rounds the regret of the algorithm is at most $m\mu^\star$. Thus, the total expected regret of the algorithm is upper bounded by:

$$\begin{aligned} m\mu^\star + \delta(n-m)\mu^\star + (1-\delta)2\epsilon(n-m) &= m\mu^\star + \delta(n-m)(\mu^\star - 2\epsilon) + 2\epsilon(n-m) \\ &\le m + (\delta + 2\epsilon)(n-m) \; . \end{aligned}$$

Recall that $\delta = 2de^{-m\epsilon^2}$. So, setting $m = \log(n-m)/\epsilon^2$ we obtain that $\delta \le 2d(n-m)^{-1}$. Overall, we obtain the regret bound

$$\log(n-m)/\epsilon^2 + 2d + 2\epsilon(n-m) \;\le\; \log(n)/\epsilon^2 + 2d + 2\epsilon n \; .$$

Finally, minimizing the above with respect to ϵ we obtain that the best choice of ϵ is

$$2\log(n)\mu^\star\epsilon^{-3} = 2n \quad \Rightarrow \quad \epsilon = \left(\frac{\log(n)}{n}\right)^{1/3} ,$$

and for this choice of ϵ we have a regret of

$$2d + 3\log^{2/3}(n)\, n^{2/3} \ .$$

This is indeed a sublinear regret. But is it optimal? In the next section we present an active algorithm that achieves an exponentially better regret rate.

2.2 UCB1

We now present an active policy for the multi-armed bandit problem (taken from the paper "Finite-time Analysis of the Multiarmed Bandit Problem" that can be found in the course website). The acronym stands for Upper Confidence Bound.

Algorithm 1 UCB1

```
Initialization
 for $t = 1, \ldots, d$
   Pull arm $y_t = t$
   Receive reward $r_t$
   Set $R_t = r_t$ and $T_t = 1$
 end for
Loop
 for $t = d+1, d+2, \ldots$
   Pull arm $y_t \in \mathrm{argmax}_j \left(\frac{R_j}{T_j} + \sqrt{\frac{2\ln(n)}{T_j}}\right)$
   Receive reward $r_t$
   Set $R_{y_t} = R_{y_t} + r_t$ and $T_{y_t} = T_{y_t} + 1$
 end for
```

The algorithm remembers the number of times each arm has been pulled and the cumulative reward obtained for each arm. Based on this information, the algorithm calculates an upper bound on the true expected reward of the arm and then it chooses the arm for which this upper bound is maximized.

To analyze UCB1, we first need a concentration measure for martingales.

Theorem 1 (Azuma) *Let $X_1, \ldots, X_n$ be a martingale (i.e. a sequence of random variables s.t. $\mathbb{E}[X_i|X_{i-1}, \ldots, X_1] = X_i$ for all $i > 1$ and $\mathbb{E}[X_1] = 0$). Assume that $|X_i - X_{i-1}| \le 1$ with probability 1. Then, for any $\epsilon > 0$ we have*

$$\mathbb{P}[|X_n| \ge n\epsilon] \le 2\exp\left(-n\epsilon^2/2\right) \ .$$

The above theorem implies:

Lemma 1 *Let $X_1, \ldots, X_n$ be a sequence of random variables over $[0, 1]$ such that $\mathbb{E}[X_i|X_{i-1}, \ldots, X_i] = \mu$ for all i. Denote $S_n = X_1 + \ldots + X_n$. Then, for any $\epsilon > 0$ we have*

$$\mathbb{P}[|S_n - n\mu| \ge n\epsilon] \le 2\exp\left(-n\epsilon^2/2\right) \ .$$

Proof For all i let $Y_i = X_1 + \ldots + X_i - i\mu$. Then,

$$\mathbb{E}[Y_{i-1}|Y_i, \ldots, Y_1] = Y_{i-1} + \mathbb{E}[X_i|X_{i-1}, \ldots, X_i] - \mu = Y_{i-1} \ .$$

Also, $|Y_i - Y_{i-1}| = |X_i - \mu| \le 1$. Applying Theorem 1 on the sequence $Y_1, \ldots, Y_n$ the proof follows. ■

The following theorem provides a regret bound for UCB1.

Theorem 2 *The regret of UCB1 is at most*

$$8\ln(n)\sum_{j\neq j^\star}\frac{1}{\Delta_j}+2\sum_{j\neq j^\star}\Delta_j\ .$$

Proof For any arm $i \neq j^\star$ denote $\Delta_i = \mu^\star - \mu_i$. The expected regret of the algorithm can be rewritten as

$$\sum_{j\neq j^\star}\Delta_j\ \mathbb{E}[T_j]\ . \tag{1}$$

In the following we will upper bound $\mathbb{E}[T_j]$.

Suppose we are on round t. We have

$$1.\quad \mathbb{P}\left[\tfrac{R_j}{T_j}-\sqrt{\tfrac{2\ln(n)}{T_j}}\geq\mu_j\right]\leq\exp(-\ln(n))=1/n \tag{2}$$

$$2.\quad \mathbb{P}\left[\tfrac{R_{j^\star}}{T_{j^\star}}+\sqrt{\tfrac{2\ln(n)}{T_{j^\star}}}\leq\mu^\star\right]\leq\exp(-\ln(n))=1/n \tag{3}$$

Therefore, with probability of at least $1-2/n$ we have that

$$\tfrac{R_j}{T_j}-\sqrt{\tfrac{2\ln(n)}{T_j}}<\mu_j=\mu^\star-\Delta_j<\tfrac{R_{j^\star}}{T_{j^\star}}+\sqrt{\tfrac{2\ln(n)}{T_{j^\star}}}-\Delta_j\ ,$$

which yields

$$\tfrac{R_j}{T_j}+\sqrt{\tfrac{2\ln(n)}{T_j}}+\left(\Delta_j-2\sqrt{\tfrac{2\ln(n)}{T_j}}\right)<\tfrac{R_{j^\star}}{T_{j^\star}}+\sqrt{\tfrac{2\ln(n)}{T_{j^\star}}}\ .$$

If $T_j \geq 8\ln(n)/\Delta_j^2$ the above implies that

$$\tfrac{R_j}{T_j}+\sqrt{\tfrac{2\ln(n)}{T_j}}<\tfrac{R_{j^\star}}{T_{j^\star}}+\sqrt{\tfrac{2\ln(n)}{T_{j^\star}}}$$

and therefore we will not pull arm j on this round with probability of at least $1-2/n$.

The above means that

$$\mathbb{E}[T_j]\leq 8\ln(n)/\Delta_j^2+\sum_t\frac{2}{n}=8\ln(n)/\Delta_j^2+2\ .$$

Combining with Eq. (1) we conclude our proof. ■

3 The exploratory multi-armed bandit problem

In the exploratory multi-armed bandit problem we seek an arm that is ϵ optimal with probability of at least $1-\delta$ (this is a PAC setup) but we do not care about the incurred regret. The question is how many samples do we need?

It is not hard to show that if we sample each arm $4/\epsilon^2\ln(2d/\delta)$ and choose the best arm we get and ϵ-optimal arm with probability of at least $1-\delta$. See Theorem 6 in "Action Elimination and Stopping Conditions for the Multi-Armed Bandit and Reinforcement Learning Problems" in the course site. It more interesting that one can get rid of the d in the log. Specifically, Theorem 10 in the above paper shows an algorithm that has a sample complexity of $O(d/\epsilon^2\ln(1/\delta)$. The idea of the proof there is to sample the arms, take the top half, sample the remaining arms and iterate.

It is perhaps a bit surprising that the result above is tight. That is, there does not exist an algorithm with PAC guarantees with sampling of less than $O(d/\epsilon^2\ln(1/\delta)$. The proof appears in "The Sample Complexity of Exploration in the Multi-Armed Bandit Problem" that can be found in the course web site. It involves a change of measure argument by showing that if the algorithm works for all arm assignments (i.e., it finds and ϵ-optimal arm with high probability) it must fail (i.e., find a wrong arm) with sufficiently high probability.