Online Learning

Introduction

- Intersection of Online Algorithms & Machine Learning
- Make decisions with limited information about the past
- Connections to Game Theory, Information Theory

Large scale applications:

- Advertisement placement
- Web ranking
- Online recommendation

Classical Machine Learning

- Batch of training examples
- Separation between training and predicting phase
- e.g. PAC learning

Online Learning

General Theme

- Regret Analysis
- Prediction from expert advice
- Multi Armed Bandits
- Noisy Models

Online Learning Model

- Sequence of consecutive rounds.
- Learner given a question and is required to provide an answer.
- Correct answer is revealed and learner suffers a loss.

Instance domain: $\mathcal{X}$
Target domain: $\mathcal{Y}$
Prediction domain: $\mathcal{D} \supseteq \mathcal{Y}$
for t=1,2,...
    receive question $x_t \in \mathcal{X}$
    predict $p_t \in \mathcal{D}$
    receive answer $y_t \in \mathcal{Y}$
    suffer loss $l(p_t, y_t)$

Learning From Examples

- Sequence of Examples
- Realizability Assumption
  - All answers are generated by some hypothesis $h^* : \mathcal{X} \to \mathcal{Y}$
  - Hypothesis (or Concept) class $\mathcal{H}$ is known to the learner.

Goal: Make as few mistakes as possible assuming h^* and $\mathcal{H}$ chosen by adversary.

Other models assume various levels of *noise* (adversarial/random).

Mistake Bound Model

Hypothesis class $\mathcal{H}$, online learning algorithm $\mathcal{A}$, integer T.

Definition (Mistake bound)

Given sequence $S = ((x_1, y_1), ..., (x_T, y_T))$, let $M_{\mathcal{A}}(S)$ be the number of mistakes algorithm $\mathcal{A}$ makes on S. Denote by $M_{\mathcal{A}}(\mathcal{H})$ the supremum of $M_{\mathcal{A}}(S)$ over all sequences of the above form. A bound of the form $M_{\mathcal{A}}(\mathcal{H}) \leq B < \infty$ is called a *mistake bound*.

Definition (Online learnability)

A hypothesis class $\mathcal{H}$ is *online learnable* if there exists an algorithm $\mathcal{A}$ for which $M_{\mathcal{A}}(\mathcal{H}) \leq B < \infty$.

Online Binary Classification

- $\mathcal{D} = \mathcal{Y} = \{0, 1\}$
- Loss function: $l(p_t, y_t) = |p_t - y_t|$.

Algorithm: **Consistent**
input: A finite hypothesis class $\mathcal{H}$
initialize: $V_1 = \mathcal{H}$
for $t = 1, 2, ..$
    receive x_t
    choose any $h \in V_t$
    predict $p_t = h(x_t)$
    receive true answer $y_t = h^*(x_t)$
    update $V_{t+1} = \{h \in V_t : h(x_t) = y_t\}$

Algorithm **Consistent** enjoys the mistake bound of
$M_{Consistent}(\mathcal{H}) \leq |\mathcal{H}| - 1$.

Halving

Algorithm: **Halving**
input: A finite hypothesis class $\mathcal{H}$
initialize: $V_1 = \mathcal{H}$
for $t = 1, 2, ..$
    receive x_t
    predict $p_t = argmax_{r \in \{0,1\}} |h \in V_t : h(x_t) = r|$
    receive true answer y_t
    update $V_{t+1} = \{h \in V_t : h(x_t) = y_t\}$

Algorithm **Halving** enjoys the mistake bound of $M_{Halving}(\mathcal{H}) \leq log_2(|\mathcal{H}|)$.

Winnow Algorithm

Let $\mathcal{H}$ be the class of monotone disjunctions over $\{0,1\}^n$.

Algorithm: **Winnow**
Initialize the weights $w_1, .., w_n$ of the variables to 1.
Given an example $x = \{x_1, .., x_n\}$
if $(w_1 x_1 + w_2 x_2 + ... + w_n x_n \geq n)$ output 1
output 0 otherwise.
If the algorithm makes a mistake:
(a) If the algorithm predicts negative on a positive example, then for each x_i equal to 1, double the value of w_i .
(b) If the algorithm predicts positive on a negative example, then for each x_i equal to 1, cut the value of w_i in half.
Repeat

The Winnow Algorithm learns the class of disjunctions in the Mistake Bound model, making at most $2 + 3r(1 + lgn)$ mistakes when the target hypothesis is a disjunction of r variables.

Online Learnability

- Littlestone's Dimension
- Standard Optimal Algorithm
- VC Dimension

Unrealizable Case

- Agnostic learning.
- Competitive with the best hypothesis in $\mathcal{H}$.
- *Regret* of the algorithm.

The regret of the algorithm relative to h when running on a sequence of T examples is defined as:

$$Regret_T(h) = \sum_{t=1}^{T} l(p_t, y_t) - \sum_{t=1}^{T} l(h(x_t), y_t)$$

The regret of the algorithm relative to a hypothesis class $\mathcal{H}$ is:

$$Regret_T(\mathcal{H}) = max_{h \in \mathcal{H}} Regret_T(h)$$

Goal: algorithm with regret sublinear in T.

Cover's Impossibility Result

- No algorithm can obtain regret sublinear in T even if $|\mathcal{H}| = 2$.

Solution: Randomize

Predicting from Expert Advice

- Learner has to choose from the advice of d given experts.
- Pay cost corresponding to the advice of the expert.
- Competitive with the cost of best fixed expert.
- Randomize choice and get expected regret.

Weighted Majority

1. Initialize the weights $w_1, .., w_n$ of all experts to 1.
2. Given set of predictions $\{x_1, ..., x_n\}$, output 1 if

$$\sum_{i:x_i=1} w_i \geq \sum_{i:x_i=0} w_i$$

   output 0 otherwise.
3. Penalize each mistaken expert by multiplying its weight by $1/2$.

Goto 2.

Theorem

The number of mistakes M made by the Weighted Majority algorithm described above is never more than $2.41(m + lg(n))$, where m is the number of mistakes made by the best expert so far.

Multi Armed Bandits

- Partial information.
- Adversarial or stochastic.
- Exploration-exploitation trade-off.

Online Convex Optimization

- Perceptron algorithm (must define correct hypothesis class)
- Convexification techniques: randomization - surrogate loss functions
- Follow-the-leader
- Online Gradient Descent