Statistical Learning Theory: Experts and Bandits

CS4780/5780 – Machine Learning
Fall 2013
Thorsten Joachims
Cornell University
Reading: Mitchell Chapter 7.5

Generalization Error Bound: Infinite H, Non-Zero Error

• Setting
  – Sample of n labeled instances S
  – Learning Algorithm L using a hypothesis space H with VCDim(H) = d
  – L returns hypothesis h_L(S) with lowest training error
• Definition: The VC-Dimension of H is equal to the maximum number d of examples that can be split into two sets in all 2^d ways using functions from H (shattering).
• Given hypothesis space H with VCDim(H) equal to d and an i.i.d. sample S of size n, with probability (1 - δ) it holds that:

\[
\text{Ferr}_{h}(S) \leq \text{Ferr}_{h}(h_L(S)) \leq \frac{ \ln \left( \frac{n}{d} \right) - 1}{\delta} + \frac{\ln \left( \frac{1}{\delta} \right)}{\delta}.
\]

Outline

• Online learning
• Review of perceptron and mistake bound
• Expert model
  – Halving Algorithm
  – Weighted Majority Algorithm
  – Exponentiated Gradient Algorithm
• Bandit model
  – EXP3 Algorithm

Online Classification Model

– Setting
  • Classification
  • Hypothesis space H with h: X → Y
  • Measure misclassifications (i.e. zero/one loss)
– Interaction Model
  • Initialize hypothesis h ∈ H
  • FOR t from 1 to T
    – Receive x_t
    – Make prediction y_t = h(x_t)
    – Receive true label y_t
    – Record if prediction was correct (e.g., y_t = y_t)
  – Update h

(Online) Perceptron Algorithm

• Input: S = ((x_1, y_1), ..., (x_n, y_n)), x_i ∈ X, y_i ∈ {-1, 1}
• Algorithm:
  – w_0 = 0, k = 0
  – FOR i = 1 to n
    – IF y_i (w_{k} · x_i) < 0
      – Make mistake
        – w_{k+1} = w_{k} + y_i x_i
      – k = k + 1
  – ENDFOR
• Output: w_k

Perceptron Mistake Bound

Theorem: For any sequence of training examples S = ((x_1, y_1), ..., (x_n, y_n)) with

\[ R = \max \| \tilde{x}_i \|, \]

if there exists a weight vector w_{\text{opt}} with \| w_{\text{opt}} \| = 1 and

\[ y_i (w_{\text{opt}} : \tilde{x}_i) \geq \delta \]

for all 1 \leq i \leq n, then the Perceptron makes at most

\[ \frac{R^2}{\delta^2} \]

errors.
Expert Learning Model

- Setting
  - $N$ experts named $H = \{h_1, \ldots, h_N\}$
  - Each expert $h_i$ takes an action $y = h_i(x_t)$ in each round $t$
    - and incurs loss $\Delta_{h_i}$
  - Algorithm can select which expert’s action to follow in each round

- Interaction Model
  - FOR $t$ from 1 to $T$
    - Algorithm selects expert $h_i$ according to strategy $A_i$ and follows its action $y$
    - Experts incur losses $\Delta_{h_1} - \Delta_{h_N}$
    - Algorithm incurs loss $\Delta_{h_i}$
    - Algorithm updates $w_t$ to $w_{t+1}$ based on $\Delta_{h_1} - \Delta_{h_N}$

Halving Algorithm

- Setting
  - $N$ experts named $H = \{h_1, \ldots, h_N\}$
  - Binary actions $y = \{+1, -1\}$ given input $x_t$, zero/one loss
  - Perfect expert exists in $H$

- Algorithm
  - $V_{S_1} = H$
  - FOR $t = 1$ TO $T$
    - Predict the same $y$ as majority of $h_i \in V_{S_t}$
    - $V_{S_{t+1}} = V_{S_t}$ minus those $h_i \in V_{S_t}$ that were wrong

Weighted Majority Algorithm

- Setting
  - $N$ experts named $H = \{h_1, \ldots, h_N\}$
  - Binary actions $y = \{+1, -1\}$ given input $x_t$, zero/one loss
  - There may be no expert in $H$ that acts perfectly

- Algorithm
  - Initialize $w_1 = (1, 1, \ldots, 1)$
  - FOR $t = 1$ TO $T$
    - Predict the same $y$ as majority of $h_i \in H_t$ each weighted by $w_t$
      - FOR EACH $h_i \in H_t$
        - IF $h_i$ incorrect THEN $w_{t+1} = w_t * 0$
          - ELSE $w_{t+1} = w_t$
  - Mistake Bound
    - How close is the number of mistakes the Weighted Majority Algorithm makes to the number of mistakes of the best expert in hindsight?

Regret

- Idea
  - Compare performance to best expert in hindsight

- Regret
  - Expected loss of algorithm $A_w$ at time $t$ is $E_{A_w}[\Delta_i] = \sum_{t=1}^T \Delta_i$
    - for randomized algorithm that picks recommendation of expert $i$ at time $t$ with probability $w_t$
  - Overall loss of best expert $\mathcal{L}^*$ in hindsight is $\sum_{t=1}^T \Delta_t, \mathcal{L}^*$

- Regret is difference between expected loss of algorithm and best fixed expert in hindsight
  - $\text{Regret}(T) = \sum_{t=1}^T w_t \Delta_t - \min_{i \in [1, N]} \sum_{t=1}^T \Delta_{t, i}, \mathcal{L}^*$

Exponentiated Gradient Algorithm for Expert Setting (EG)

- Setting
  - $N$ experts named $H = \{h_1, \ldots, h_N\}$
  - Any actions, any loss function
    - There may be no expert in $H$ that acts perfectly

- Algorithm
  - Initialize $w_1 = \left(\frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N}\right)$
  - FOR $t$ from 1 to $T$
    - Algorithm randomly picks $i_t$ from $P(i_t = i_t) = w_t$
      - Experts incur losses $\Delta_{i_t} - \Delta_{i_t}$
    - Algorithm incurs loss $\Delta_{i_t}$
      - Algorithm updates $w_t$ for all experts $i_t$ as $w_t(i_t) = w_t(i_t) \exp(-\eta \Delta_{i_t})$
      - Then normalize $w_t$ so that $\sum_{i_t} w_t(i_t) = 1.$

Regret Bound for Exponentiated Gradient Algorithm

- Theorem
  - The regret of the exponentiated gradient algorithm in the expert setting is bounded by
    - $\text{Regret}(T) \leq \Delta \sqrt{2T \log(N)}$
  - where $\Delta = \max \{\Delta_{i, t}\}$ and $\eta = \frac{1}{\Delta \sqrt{2T}}$.
Bandit Learning Model

- Setting
  - $N$ bandits named $H = \{h_1, ..., h_N\}$
  - Each bandit $h_i$ takes an action in each round $t$ and incurs loss $\Delta_{i,t}$
  - Algorithm can select which bandit’s action to follow in each round

- Interaction Model
  - FOR $t$ from 1 to $T$
    - Algorithm selects expert $h_i$ according to strategy $A_w$ and follows its action $y$
    - Bandits incur losses $\Delta_{i,1} - \Delta_{i,N}$
    - Algorithm incurs loss $\Delta_{t,i}$
    - Algorithm updates $w_i$ to $w_{i+1}$ based on $\Delta_{i,t}$

Key difference compared to Expert Model

Exponentiated Gradient Algorithm for Bandit Setting (EXP3)

- Initialize $w_1 = \left(\frac{1}{N}, ..., \frac{1}{N}\right)$, $y = \min \left\{1, \frac{N \log N}{(e-1)\Delta T}\right\}$
- FOR $t$ from 1 to $T$
  - Algorithm randomly picks $i_t$ with probability
    $P(i_t) = (1 - y)w_{i_t} + y/N$
  - Experts incur losses $\Delta_{t,1} - \Delta_{t,N}$
  - Algorithm incurs loss $\Delta_{t,i}$
  - Algorithm updates $w$ for bandit $i_t$ as
    $w_{t+1,i_t} = w_{t,i_t} \exp \left(-\eta \Delta_{t,i_t}/P(i_t)\right)$
  - Then normalize $w_{t+1}$ so that $\sum_j w_{t+1,j} = 1.$

Other Online Learning Problems

- Stochastic Experts
- Stochastic Bandits
- Online Convex Optimization
- Partial Monitoring