Statistical Learning Theory: Weighted Experts and Bandits
CS4780/5780 – Machine Learning
Fall 2014
Thorsten Joachims
Cornell University
Reading: Mitchell Chapter 7.5

Setting

- Setting
  - N experts named \( H = \{ h_1, ..., h_N \} \)
  - Each expert \( h_i \) takes an action \( y = h_i(x_t) \) in each round \( t \) and incurs loss \( \Delta_{i,t} \)
  - 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 \( A(w_t) \) and follows its action \( y \)
  - Experts incur losses \( \Delta_{i,1} - \Delta_{i,N} \)
  - Algorithm updates \( w_t \) to \( w_{t+1} \) based on \( \Delta_{i,1} - \Delta_{i,N} \)

Expert Learning Model

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

• Algorithm
  - Initialize \( w_1 = (1,1, ..., 1) \)
  - FOR \( t = 1 \) TO \( T \)
    - Predict the same \( y \) as majority of \( h_i \) in \( H \), each weighted by \( w_t \)
      - IF \( h_i \) unanimous THEN \( w_{t+1} = w_t \) (tie)
      - ELSE \( w_{t+1} = w_t \cdot y \)
  - 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?

Weighted Majority Algorithm

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

• Algorithm
  - Initialize \( w_1 = (1,1, ..., 1) \)
  - FOR \( t = 1 \) TO \( T \)
    - Predict the same \( y \) as majority of \( h_i \) in \( H \), each weighted by \( w_t \)
      - IF \( h_i \) unanimous THEN \( w_{t+1} = w_t \) (tie)
      - ELSE \( w_{t+1} = w_t \cdot y \)
  - 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?

Expected Regret

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

• Algorithm
  - Initialize \( w_1 = \left( \frac{1}{N}, ..., \frac{1}{N} \right) \)
  - FOR \( t \) from 1 to \( T \)
    - Algorithm randomly picks \( i_t \) from \( P(i_t = i) = w_t \)
    - Experts incur losses \( \Delta_{i,1} - \Delta_{i,N} \)
    - Algorithm updates \( w_t \) for all experts \( i \) as
      \[ w_{t+1} = w_t \cdot \exp(-\eta \Delta_{i_t}) \]
      Then normalize \( w_{t+1} \) so that \( \sum_i w_{t+1} = 1 \).

Regret

- Idea
  - \( N \) experts named \( H = \{ h_1, ..., h_N \} \)
  - Compare performance of \( A \) to best expert \( i^* \) in hindsight.

• Regret
  - Overall loss of best expert \( i^* \) in hindsight is
    \[ \Delta_{i^*} = \min_{i \in \{1, ..., N\}} \sum_{t=1}^T \Delta_{i,t} \]
  - Loss of algorithm \( A \) at time \( t \) is
    \[ \Delta_{A} = \sum_{t=1}^T \Delta_{A(t),t} \]
  - Regret is difference between loss of algorithm and best fixed expert in hindsight
    \[ \text{Regret}(T) = \sum_{t=1}^T \Delta_{A(t),t} - \min_{i \in \{1, ..., N\}} \sum_{t=1}^T \Delta_{i,t} \]

Exponentiated Gradient Algorithm for Expert Setting (EG)

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

• Algorithm
  - Initialize \( w_1 = \left( \frac{1}{N}, ..., \frac{1}{N} \right) \)
  - FOR \( t \) from 1 to \( T \)
    - Algorithm randomly picks \( i_t \) from \( P(i_t = i) = w_t \)
    - Experts incur losses \( \Delta_{i,1} - \Delta_{i,N} \)
    - Algorithm updates \( w_t \) for all experts \( i \) as
      \[ w_{t+1} = w_t \cdot \exp(-\eta \Delta_{i_t}) \]
      Then normalize \( w_{t+1} \) so that \( \sum_i w_{t+1} = 1 \).
Regret Bound for Exponentiated Gradient Algorithm

• Theorem
  The expected regret of the exponentiated gradient algorithm in the expert setting is bounded by
  \[
  \text{Expected Regret}(T) \leq \Delta^\text{max} \sqrt{2T \log |H|}
  \]
  where \(\Delta^\text{max} = \max \{\Delta_t, i\}\) and
  \[
  \eta = \frac{\sqrt{\log(N)}}{\Delta^2 T}.
  \]

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 \(\Delta_t, i\)
  – 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 \(A_{w_t}\) and follows its action \(y\)
    • Bandits incur losses \(\Delta_t, 1 \ldots \Delta_t, N\)
    • Algorithm incurs loss \(\Delta_t, i\)
    • Algorithm updates \(w_t\) to \(w_{t+1}\) based on \(\Delta_t, i\)

Other Online Learning Problems

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

Exponentiated Gradient Algorithm for Bandit Setting (EXP3)

• Initialize \(w_1 = (1/N, \ldots, 1/N), y = \min \left\{1, \frac{N \log N}{(e-1)\Delta T}\right\}\)

• FOR \(t\) from 1 to \(T\)
  – Algorithm randomly picks \(i_t\) with probability \(P(i_t) = (1-y)w_t,i + y/N\)
  – Experts (aka Bandits) incur losses \(\Delta_t, 1 \ldots \Delta_t, N\)
  – Algorithm incurs loss \(\Delta_t, i_t\)
  – Algorithm updates \(w_t\) for bandit \(i_t\) as
    \[
    w_{t+1, i_t} = w_{t, i_t} \exp(-\eta \Delta_t, i_t / P(i_t))
    \]
    Then normalize \(w_{t+1}\) so that \(\sum_j w_{t+1, j} = 1\).