Efficient Bandit Algorithms for Online Multiclass Prediction

Sham M. Kakade
Shai Shalev-Shwartz
Ambuj Tewari

ICML, July 2008
Motivation

- Online web advertisement systems
  - User submits a query
  - System (the learner) places an ad
  - User either “clicks” or ignores
  - Goal: Maximize number of “clicks”

Modeling ?

- Not the common online learning setting --
  If user ignores, we don’t get the “correct” ad

- Not the common multi-armed bandit --
  We are also provided with a query
Outline

- Online Bandit Multi-class Categorization
- Background: The Multi-class Perceptron
- The Banditron
- Analysis
- Experiments
- The Separable Case
- Extensions and Open Problems
Online Bandit Multiclass Categorization

For $t = 1, 2, \ldots, T$

- Receive $x \in \mathbb{R}^d$ (query)
- Predict $\hat{y}_t \in \{1, \ldots, k\}$ (ad)
- Pay $1[y_t \neq \hat{y}_t]$ (click feedback)
- $y_t$ is not revealed
A hypothesis is a mapping $h : \mathbb{R}^d \rightarrow \{1, \ldots, k\}$

Linear hypothesis: Exists $k \times d$ matrix $W$ s.t.

$$h(x) = \arg\max_r (W x)_r$$
The Multiclass Perceptron

For $t = 1, 2, \ldots, T$

- Receive $x_t \in \mathbb{R}^d$
- Predict $\hat{y}_t = \operatorname{argmax}_r (W^t x_t)_r$
- Receive $y_t$
- Update: $W^{t+1} = W^t + U^t$ where $U^t = \begin{bmatrix} 0 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & 0 \end{bmatrix}$

row $\hat{y}_t$

row $y_t$
Perceptron in the Bandit Setting

\[ U^t = \begin{bmatrix} 0 & \ldots & 0 \\ \vdots \\ 0 & \ldots & x_t & \ldots \\ \ldots & x_t & \ldots \\ 0 & \ldots & 0 \\ \vdots \\ 0 & \ldots & \ldots & \ldots \\ \ldots & \ldots & 0 \\ 0 & \ldots & 0 \\ \vdots \\ 0 & \ldots & 0 \end{bmatrix} \]

- **Problem:** We’re blind to value of \( y_t \)
- **Solution:** Randomization can help!
- **Explore:** instead of predicting $\hat{y}_t$ guess some $\tilde{y}_t$

- Suppose we get the feedback 'correct', i.e. $\tilde{y}_t = y_t$

- Then, we know that
  - $\hat{y}_t \neq y_t$
  - $y_t = \tilde{y}_t$

- So, we can update $W$ using the matrix $U^t$
Exploration vs. Exploitation

- But, if our current model is correct, i.e. $\hat{y}_t = y_t$
- And, we guess some other $\tilde{y}_t$
- Then, we both suffer loss and do not know how to update $W$
- In this case, it’s better to **Exploit** the quality of current model
- We control the **exploration-exploitation tradeoff** using randomization
The Banditron

For $t = 1, 2, \ldots, T$

- Receive $x_t \in \mathbb{R}^d$

- Set $\hat{y}_t = \arg\max_r (W^t x_t)_r$

- Define: $P(r) = (1 - \gamma) 1[r = \hat{y}_t] + \frac{\gamma}{k}$

- Randomly sample $\tilde{y}_t$ according to $P$

- Predict $\tilde{y}_t$ and receive feedback $1[\tilde{y}_t = y_t]$

- Update: $W^{t+1} = W^t + \tilde{U}^t$
For $t = 1, 2, \ldots, T$

- Receive $\mathbf{x}_t \in \mathbb{R}^d$
- Set $\hat{y}_t = \text{argmax}_r (W^t \mathbf{x}_t)_r$
- Define: $P(r) = (1 - \gamma)1[r = \hat{y}_t] + \gamma k$
- Randomly sample $\tilde{y}_t$ according to $P$
- Predict $\tilde{y}_t$ and receive feedback $1[\tilde{y}_t = y_t]$
- Update: $W^{t+1} = W^t + \tilde{U}^t$
The Banditron Expected Update

\[ \mathbb{E}[\tilde{U}^t] = \sum_r P(r) \begin{bmatrix} 0 & \ldots & 0 \\ \vdots \\ 0 & \ldots & 0 \\ \ldots & \frac{1[y_t=r]}{P(y_t)} x_t & \ldots \\ 0 & \ldots & 0 \\ \ldots & -x_t & \ldots \\ 0 & \ldots & 0 \\ \vdots \\ 0 & \ldots & 0 \end{bmatrix} = U^t \]
Analysis: The Hinge-Loss

\[ \ell_t(W) = \max_{r \neq y_t} 1 - (W x_t)_{y_t} + (W x_t)_r \geq 1[y_t \neq \hat{y}_t] \]
Analysis: The Hinge-Loss

\[ \ell_t(W) = \max_{r \neq y_t} 1 - (W x_t)_y + (W x_t)_r \geq 1[y_t \neq \hat{y}_t] \]

The Separable Case:
Mistake Bounds

Perceptron:

\[ M \leq L + D + \sqrt{LD} \]

Banditron:

\[ \mathbb{E}[M] \leq L + \gamma T + 3 \max \left\{ \frac{kD}{\gamma}, \sqrt{D \gamma T} \right\} + \sqrt{\frac{kD L}{\gamma}} \]

---

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M )</td>
<td># mistakes</td>
</tr>
<tr>
<td>( L )</td>
<td>competitor loss ( \sum_t \ell_t(W^*) )</td>
</tr>
<tr>
<td>( D )</td>
<td>competitor margin ( |W^*|_F^2 )</td>
</tr>
<tr>
<td>( k )</td>
<td># classes</td>
</tr>
<tr>
<td>( T )</td>
<td># rounds</td>
</tr>
<tr>
<td>( \gamma )</td>
<td>Exploration-Exploitation parameter</td>
</tr>
</tbody>
</table>
### Mistake Bounds (cont.)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(L)</td>
<td>competitor loss (\sum_t \ell_t(W^*))</td>
</tr>
<tr>
<td>(D)</td>
<td>competitor margin (|W^*|_F^2)</td>
</tr>
<tr>
<td>(k)</td>
<td># classes</td>
</tr>
<tr>
<td>(T)</td>
<td># rounds</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No noise: (L = 0)</th>
<th>Perceptron</th>
<th>Banditron</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(D)</td>
<td>(\sqrt{kDT})</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Low noise: (L = O(\sqrt{kDT}))</th>
<th>Perceptron</th>
<th>Banditron</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(\sqrt{kDT})</td>
<td>(\sqrt{kDT})</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Noisy:</th>
<th>Perceptron</th>
<th>Banditron</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(L + T^{1/2})</td>
<td>(L + T^{2/3})</td>
</tr>
</tbody>
</table>
Experiments

- **Reuters RCV1**
  - ~700k documents
  - Bag-of-words (d ~ 350k)
  - 4 labels \{CCAT, ECAT, GCAT, MCAT\}

- **Synthetic separable data set**
  - 9 classes, d=400, million instances
  - A simple simulation of generating text documents

- **Synthetic non-separable data set**
  - separable + 5% label noise
Experimental Results – Reuters

\[ \gamma = 0.050 \]

Error rate vs. number of examples for Perceptron and Banditron with similar slope.
Experimental Results – Separable Data

\[ \gamma = 0.014 \]

The graph shows the error rate versus the number of examples for different algorithms. The lines indicate the following slopes:
- Slope -1
- Slope -0.55

The error rate is plotted on a logarithmic scale, and the number of examples is also plotted on a logarithmic scale. The graph compares the performance of Perceptron and Banditron algorithms.
Experimental Results – 5% label noise

\[ \gamma = 0.006 \]

![Graph showing error rate vs. number of examples for Perceptron and Banditron with 5% label noise. The graph indicates that Banditron has a lower error rate compared to Perceptron.]

- Perceptron error rate: 13%
- Banditron error rate: 10%
Exploration-Exploitation Parameter

5% label noise

- Perceptron
- Banditron

Reuters
The Separable Case

Halving

- Discretized hypothesis space
- Predict by majority vote
- Remove 'wrong' hypotheses
- Note: can be applied in Bandit setting
- Mistake Bound $O(k^2 d \log(Dd))$
- Using JL lemma we can also obtain $O(k^2 D \log(\frac{T+k}{\delta}) \log(D))$
The Separable Case

Halving

- Discretized hypothesis space
- Predict by majority vote
- Remove ’wrong’ hypotheses
- Note: can be applied in Bandit setting
- Mistake Bound $O(k^2 d \log(Dd))$
- Using JL lemma we can also obtain $O(k^2 D \log\left(\frac{T+k}{\delta}\right) \log(D))$
The Separable Case

**Halving**

- Discretized hypothesis space
- Predict by majority vote
- Remove ’wrong’ hypotheses
- Note: can be applied in Bandit setting
- Mistake Bound $O(k^2 d \log(Dd))$
- Using JL lemma we can also obtain $O(k^2 D \log(\frac{T+k}{\delta}) \log(D))$
Extensions and Open Problems

- Label Ranking
  - Predicting a “label ranking”
  - How to interpret feedback?
- Multiplicative and Margin-based updates
  - Bandit versions of “Winnow” and “Passive-Aggressive”
- Deterministic vs. Randomized strategies
- Achievable rates?
  - Efficient algorithms for the separable case?