New Algorithms for Contextual Bandits

Lev Reyzin
Georgia Institute of Technology

Work done at Yahoo!
A. Beygelzimer, J. Langford, L. Li, L. Reyzin, R.E. Schapire
Contextual Bandit Algorithms with Supervised Learning
Guarantees (AISTATS 2011)

M. Dudik, D. Hsu, S. Kale, N. Karampatziakis, J. Langford,
L. Reyzin, T. Zhang Efficient Optimal Learning for
Contextual Bandits (UAI 2011)

S. Kale, L. Reyzin, R.E. Schapire Non-Stochastic Bandit
Slate Problems (NIPS 2010)
Serving Content to Users

Query, IP address, browser properties, etc.
Serving Content to Users

Query, IP address, browser properties, etc.

result (ie. ad, news story)
Serving Content to Users

Query, IP address, browser properties, etc.

result (ie. ad, news story)

click or not
Serving Content to Users

Query, IP address, browser properties, etc.

result (ie. ad, news story)

click or not
Serving Content to Users

Query, IP address, browser properties, etc.

result (ie. ad, news story)

click or not
Serving Content to Users

context $x_t$

action $j_t$

reward $r_{jt}(t)$
Outline

- The setting and some background
- Show ideas that fail
- Give a high probability optimal algorithm
- Dealing with VC sets
- An efficient algorithm
- Slates
Multiarmed Bandits
[Robbins ’52]

$0$
Multiarmed Bandits
[Robbins ’52]
Multiarmed Bandits
[Robbins ’52]
Multiarmed Bandits

[Robbins ’52]

$0.50 \quad \$0.33 \quad \$0.83$

$1 \quad 2 \quad 3 \quad 4 \quad \ldots \quad T$

$1 \quad 2 \quad 3 \quad \ldots \quad k$
Multiarmed Bandits
[Robbins ’52]

1
2
3
4
...
T

$0.4T

1
$0.50

2
$0.20

3
$0.33

$0

k
$0.90
Multiarmed Bandits
[Robbins ’52]

1 \rightarrow $0.5T
2 \rightarrow $0.2T
3 \rightarrow $0.33T
\ldots
k \rightarrow $0.1T
Multiarmed Bandits

[Robbins ’52]

regret = $0.1T

$0.5T \rightarrow 1

$0.2T \rightarrow 2

$0.33T \rightarrow 3

\ldots

$0.1T \rightarrow k

$0.4T
Contextual Bandits

[Auer-CesaBianchi-Freund-Schapire ’02]

context:

1 2 3 4 ...

T

N experts/policies/functions think of N >> K
Contextual Bandits
[Auer-CesaBianchi-Freund-Schapire ’02]

context: $x_1$

$1 \ 2 \ 3 \ 4 \ \ldots \ \ T$

$1 \ 2 \ 3 \ 4 \ \ldots \ \ T$

N experts/policies/functions think of $N >> K$
Contextual Bandits
[Auer-CesaBianchi-Freund-Schapire ’02]

context: \( x_1 \)

1 $0.15

2

3

\vdots

k

N experts/policies/functions think of \( N \gg K \)
Contextual Bandits
[Auer-CesaBianchi-Freund-Schapire ’02]

N experts/policies/functions think of $N \gg K$

**context:** $x_1, x_2, x_3, x_4, \ldots, x_T$

1. $x_1$: $1$, $0.15$
2. $x_2$: $2$, $0.5$
3. $x_3$: $3$, $0.2$
4. $x_4$: $4$, $0.1$
5. $\vdots$
6. $x_T$: $T$, $0.2$

Total: $0.2T$

$1, 2, 3, 4, \ldots, k$

$0.12T$, $0.22T$, $0.1T$, $0.2T$, $0.17T$, $0$, $0$
Contextual Bandits
[Auer-CesaBianchi-Freund-Schapire ’02]

context: $x_1, x_2, x_3, x_4, \ldots, x_T$

regret = $0.02T$

N experts/policies/functions think of $N \gg K$
Contextual Bandits
[Auer-Cesa-Bianchi-Freund-Schapire ’02]

Context: $x_1, x_2, x_3, x_4, \ldots, x_T$

1. The rewards can come i.i.d. from a distribution or be arbitrary stochastic / adversarial.

2. The experts can be present or not. contextual / non-contextual.
The Setting

- T rounds, K possible actions, N policies $\pi$ in $\Pi$ (context $\rightarrow$ actions)
- for $t=1$ to $T$
  - world commits to rewards $r(t) = r_1(t), r_2(t), \ldots, r_K(t)$ (adversarial or iid)
  - world provides context $x_t$
  - learner's policies recommend $\pi_1(x_t), \pi_2(x_t), \ldots, \pi_N(x_t)$
  - learner chooses action $j_t$
  - learner receives reward $r_{j_t}(t)$
- want to compete with following the best policy in hindsight
Regret

- **Reward of algorithm A:**
  \[ G_A(T) = \sum_{t=1}^{T} r_{j_t}(t) \]

- **Expected reward of policy i:**
  \[ G_i(T) = \sum_{t=1}^{T} \pi_i(x_t) \cdot r(t) \]

- **Algorithm A’s regret:**
  \[ \max_i G_i(T) - G_A(T) \]
Regret

- **algorithm A’s regret:** \( \max_i G_i(T) - G_A(T) \)

- **bound on expected regret:** \( \max_i G_i(T) - E[G_A(T)] < \varepsilon \)

- **high probability bound:** \( P[\max_i G_i(T) - G_A(T) > \varepsilon] \leq \delta \)
Harder than supervised learning:
- In the bandit setting we do not know the rewards of actions not taken.

Many applications
- Ad auctions, medicine, finance, ...

Exploration/Exploitation
- Can exploit expert/arm you’ve learned to be good.
- Can explore expert/arm you’re not sure about.
Some Barriers

$\Omega(kT)^{1/2}$ (non-contextual) and $\sim \Omega(TK \ln N)^{1/2}$ (contextual) are known lower bounds [Auer et al. ’02] on regret, even in the stochastic case.

Any algorithm achieving regret $\tilde{O}(KT \text{ polylog } N)^{1/2}$ is said to be optimal.

$\varepsilon$-greedy algorithms that first explore (act randomly) and then exploit (follow the best policy) cannot be optimal. Any optimal algorithm must be adaptive.
Two Types of Approaches

Algorithm: at every time step
1) pull arm with highest UCB
2) update confidence bound of the arm pulled.

Algorithm: at every time step
1) sample from distribution defined by weights (mixed w/ uniform)
2) update weights “exponentially”
UCB vs EXP3
A Comparison

UCB
[Auer ’02]

◆ Pros
◆ Optimal for the stochastic setting.
◆ Succeeds with high probability.

◆ Cons
◆ Does not work in the adversarial setting.
◆ Is not optimal in the contextual setting.

EXP3 & Friends
[Auer-CesaBianchi-Freund-Schapire ’02]

◆ Pros
◆ Optimal for both the adversarial and stochastic settings.
◆ Can be made to work in the contextual setting

◆ Cons
◆ Does not succeed with high probability in the contextual setting (only in expectation).
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Regret</th>
<th>High Prob?</th>
<th>Context?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exp4 [ACFS ’02]</td>
<td>$\tilde{O}(KT \ln(N))^{1/2}$</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>$\varepsilon$ -greedy, epoch-greedy [LZ ’07]</td>
<td>$\tilde{O}((K \ln(N)^{1/3})T^{2/3})$</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Exp3.P[ACFS ’02] UCB [Auer ’00]</td>
<td>$\tilde{O}(KT)^{1/2}$</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

$\Omega\left(\sqrt{KT}\right)$ lower bound [ACFS ’02]
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Regret</th>
<th>High Prob?</th>
<th>Context?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exp4 [ACFS ’02]</td>
<td>$\tilde{O}(KT \ln(N))^{1/2}$</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>$\varepsilon$ -greedy, epoch-geedy [LZ ’07]</td>
<td>$\tilde{O}((K \ln(N)^{1/3})T^{2/3})$</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Exp3.P [ACFS ’02]</td>
<td>$\tilde{O}(KT)^{1/2}$</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>UCB [Auer ’00]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exp4.P [BLLRS ’10]</td>
<td>$\tilde{O}(K \ln(N/\delta)T)^{1/2}$</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

$\Omega\left(\sqrt{KT}\right)$ lower bound [ACFS ’02]
EXP4P
[Beygelzimer-Langford-Li-R-Schapire ’11]

Main Theorem [Beygelzimer-Langford-Li-R-Schapire ’11]: For any $\delta > 0$, with probability at least $1 - \delta$, EXP4P has regret at most $O(KT \ln (N/\delta))^{1/2}$ in the adversarial contextual bandit setting.

EXP4P combines the advantages of Exponential Weights and UCB, optimal for both the stochastic and adversarial settings works for the contextual case (and also the non-contextual case) a high probability result
Outline

- The setting and some background
- Show ideas that fail
- Give a high probability optimal algorithm
- Dealing with VC sets
- An efficient algorithm
- Slates
Bad idea 1: Maintain a set of plausible hypotheses and randomize uniformly over their predicted actions.
**Bad idea 1:** Maintain a set of plausible hypotheses and randomize uniformly over their predicted actions.

- Adversary has two actions, one always paying off 1 and the other 0. Hypothesis generally agree on correct action, except for a different one which defects each round. This incurs regret of \( \sim T/2 \).
Some Failed Approaches

- **Bad idea 1**: Maintain a set of plausible hypotheses and randomize uniformly over their predicted actions.
  - Adversary has two actions, one always paying off 1 and the other 0. Hypothesis generally agree on correct action, except for a different one which defects each round. This incurs regret of \(~T/2\).

- **Bad idea 2**: Maintain a set of plausible hypotheses and randomize uniformly among the hypothesis.
Some Failed Approaches

- **Bad idea 1**: Maintain a set of plausible hypotheses and randomize uniformly over their predicted actions.
  - Adversary has two actions, one always paying off 1 and the other 0. Hypothesis generally agree on correct action, except for a different one which defects each round. This incurs regret of $\sim T/2$.

- **Bad idea 2**: Maintain a set of plausible hypotheses and randomize uniformly among the hypothesis.
  - Adversary has two actions, one always paying off 1 and the other 0. If all but one of $> 2T$ hypothesis always predict wrong arm, and only 1 hypothesis always predicts good arm, with probability $> 1/2$ it is never picked and algorithm incurs regret of $T$. 
Rough idea of \( \epsilon \)-greedy (or \( \epsilon \)-first): act randomly for \( \epsilon \) rounds, then go with best (arm or expert).

Even if we know the number of rounds in advance, \( \epsilon \)-first won’t get us regret \( O(T)^{1/2} \), even in the non-contextual setting.

Rough analysis: even for just 2 arms, we suffer regret of \( \epsilon + (T- \epsilon)/(\epsilon^{1/2}) \).
- \( \epsilon \approx T^{2/3} \) is optimal tradeoff.
- gives regret \( \approx T^{2/3} \)
Outline

- The setting and some background
- Show ideas that fail
- **Give a high probability optimal algorithm**
- Dealing with VC sets
- An efficient algorithm
- Slates
Ideas Behind Exp4.P
(all appeared in previous algorithms)

- exponential weights
  - keep a weight on each expert that drops exponentially in the expert’s (estimated) performance

- upper confidence bounds
  - use an upper confidence bound on each expert’s estimated reward

- ensuring exploration
  - make sure each action is taken with some minimum probability

- importance weighting
  - give rare events more importance to keep estimates unbiased
Exponential Weight Algorithm for Exploration and Exploitation with Experts

(EXP4) [Auer et al. ’95]

Initialization: \( \forall \pi \in \Pi : w_t(\pi) = 1 \)

For each \( t = 1, 2, \ldots \):

1. Observe \( x_t \) and let for \( a = 1, \ldots, K \)
   \[
   p_t(a) = (1 - K p_{\text{min}}) \frac{\sum_{\pi} 1[\pi(x_t) = a] w_t(\pi)}{\sum_{\pi} w_t(\pi)} + p_{\text{min}},
   \]
   where \( p_{\text{min}} = \sqrt{\frac{\ln |\Pi|}{KT}} \).

2. Draw \( a_t \) from \( p_t \), and observe reward \( r_t(a_t) \).

3. Update for each \( \pi \in \Pi \)
   \[
   w_{t+1}(\pi) = \begin{cases} 
   w_t(\pi) \exp \left( p_{\text{min}} \frac{r_t(a_t)}{p_t(a_t)} \right) & \text{if } \pi(x_t) = a_t \\
   w_t(\pi) & \text{otherwise}
   \end{cases}
   \]
Exponential Weight Algorithm for Exploration and Exploitation with Experts

(Exp4.P) [Beygelzimer, Langford, Li, R, Schapire ’10]

Initialization: \( \forall \pi \in \Pi : w_t(\pi) = 1 \)

For each \( t = 1, 2, \ldots \):

1. Observe \( x_t \) and let for \( a = 1, \ldots, K \)
   \[
   p_t(a) = (1 - Kp_{\text{min}}) \frac{\sum_\pi 1[\pi(x_t) = a] w_t(\pi)}{\sum_\pi w_t(\pi)} + p_{\text{min}},
   \]
   where \( p_{\text{min}} = \sqrt{\frac{\ln |\Pi|}{KT}} \).

2. Draw \( a_t \) from \( p_t \), and observe reward \( r_t(a_t) \).

3. Update for each \( \pi \in \Pi \)
   \[
   w_{t+1}(\pi) = w_t(\pi) \exp \left( \frac{p_{\text{min}}}{2} \left( 1[\pi(x_t) = a_t] \frac{r_t(a_t)}{p_t(a_t)} + \frac{1}{p_t(\pi(x_t))} \sqrt{\frac{\ln N/\delta}{KT}} \right) \right).
   \]
Lemma 1

The estimated reward of an expert is \( \hat{G}_i \doteq \sum_{t=1}^{T} \hat{y}_i(t) \).

We also define \( \hat{\sigma}_i \doteq \sqrt{KT} + \frac{1}{\sqrt{KT}} \sum_{t=1}^{T} \hat{\nu}_i(t) \).

Lemma \( \Pr \left[ \exists i : G_i \geq \hat{G}_i + \sqrt{\ln(N/\delta)} \hat{\sigma}_i \right] \leq \delta \).

Proof uses a new Freedman-style martingale inequality.
Lemma 2

\[ \hat{U} = \max_i \left( \hat{G}_i + \hat{\sigma}_i \cdot \sqrt{\ln(N/\delta)} \right). \]

\[ G_{\text{Exp4.P}} \geq \left( 1 - 2\sqrt{\frac{K \ln N}{T}} \right) \hat{U} - 2\sqrt{KT \ln(N/\delta)} - \sqrt{KT \ln N} - \ln(N/\delta). \]

Proof tracks the weights of experts, similar to Exp4.

Lemmas 1 and 2 imply: \[ G_{\text{Exp4.P}} \geq G_{\text{max}} - 6\sqrt{KT \ln(N/\delta)}. \]
**Main Theorem** [Beygelzimer-Langford-Li-R-Schapire ’11]: For any $\delta > 0$, with probability at least $1 - \delta$, EXP4P has regret at most $O(KT \ln (N/\delta))^{1/2}$ in the adversarial contextual bandit setting.

**Key insights (on top of UCB/EXP):**

1) exponential weights and upper confidence bounds “stack”
2) generalized Bernstein’s inequality for martingales
## Efficiency

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Regret</th>
<th>High Prob?</th>
<th>Context?</th>
<th>Efficient?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exp4 [ACFS ’02]</td>
<td>$\tilde{O}(T^{1/2})$</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>epoch-geedy [LZ ’07]</td>
<td>$\tilde{O}(T^{2/3})$</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Exp3.P/UCB [ACFS ’02][A ’00]</td>
<td>$\tilde{O}(T^{1/2})$</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Exp4.P [BLLRS ’10]</td>
<td>$\tilde{O}(T^{1/2})$</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
EXP4P Applied to Yahoo!

Monday, February 27, 2012

Angelina Jolie mocked for flashing thigh
The actress finds herself the butt of an Oscar winner's joke after striking a revealing pose onstage. Watch >>

TRENDING NOW
01 Lucy Lawless
02 Amanda Seyfried
03 Kim Dotcom
04 Jessica Chastain
05 Oscar winners
06 Jonah Hill
07 Oscars
08 Gay marriage
09 Stock market
10 Smartphones

Now, truly Unlimited data for your iPhone.*
Get it now

Who pays to raise campaign money?
You, as a taxpayer, split the cost with the Obama campaign when the president travels to fundraisers.
We chose a policy class for which we could efficiently keep track of the weights.

- Created 5 clusters, with users (at each time step) getting features based on their distances to clusters.
- Policies mapped clusters to article (action) choices.
- Ran on personalized news article recommendations for Yahoo! front page.

We used a learning bucket on which we ran the algorithms and a deployment bucket on which we ran the greedy (best) learned policy.
Experimental Results

 Reported estimated (normalized) click-through rates on front page news. Over 41M user visits. 253 total articles. 21 candidate articles per visit.

<table>
<thead>
<tr>
<th></th>
<th>EXP4P</th>
<th>EXP4</th>
<th>ε-greedy</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Learning eCTR</strong></td>
<td>1.0525</td>
<td>1.0988</td>
<td>1.3829</td>
</tr>
<tr>
<td><strong>Deployment eCTR</strong></td>
<td><strong>1.6512</strong></td>
<td>1.5309</td>
<td>1.4290</td>
</tr>
</tbody>
</table>
Experimental Results

Reported estimated (normalized) click-through rates on front page news. Over 41M user visits. 253 total articles. 21 candidate articles per visit.

<table>
<thead>
<tr>
<th></th>
<th>EXP4P</th>
<th>EXP4</th>
<th>$\varepsilon$-greedy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning eCTR</td>
<td>1.0525</td>
<td>1.0988</td>
<td>1.3829</td>
</tr>
<tr>
<td>Deployment eCTR</td>
<td>1.6512</td>
<td>1.5309</td>
<td>1.4290</td>
</tr>
</tbody>
</table>

Why does this work in practice?
Outline

- The setting and some background
- Show ideas that fail
- Give a high probability optimal algorithm
- **Dealing with VC sets**
- An efficient algorithm
- Slates
What if we have an infinite number of policies?

Our bound of $\tilde{O}(K \ln(N)T)^{1/2}$ becomes vacuous.

If we assume our policy class has a finite VC dimension $d$, then we can tackle this problem.

Need i.i.d. assumption. We will also assume $k=2$ to illustrate the argument.
The **VC dimension** of a hypothesis class captures the class’s expressive power.

It is the cardinality of the largest set (in our case, of contexts) the class can shatter.

**Shatter** means to label in all possible configurations.
The VE algorithm:

- Act uniformly at random for $\tau$ rounds.
- This partitions our policies $\Pi$ into equivalence classes according to their labelings of the first $\tau$ examples.
- Pick one representative from each equivalence class to make $\Pi'$. 
- Run Exp4.P on $\Pi'$. 

VE, an Algorithm for VC Sets
Sauer’s lemma bounds the number of equivalence classes to \((e \tau /d)^d\).

Hence, using Exp4.P bounds, VE’s regret to \(\Pi’\) is 

\[\approx \tau + O(Td \ln(\tau))\]

We can show that the regret of \(\Pi’\) to \(\Pi\) is \(\approx (T/\tau)(d \ln T)\)

by looking at the probability of disagreeing on future data given agreement for \(\tau\) steps.

\[\tau \approx (Td \ln 1/\delta)^{1/2}\] achieves the optimal trade-off.

Gives \(\tilde{O}(Td)^{1/2}\) regret.

Still inefficient!
Outline

- The setting and some background
- Show ideas that fail
- Give a high probability optimal algorithm
- Dealing with VC sets
- An efficient algorithm
- Slates
For EXP4P, the dependence on $N$ in the regret is logarithmic.

this suggests

We could compete with a large, even super-polynomial number of policies! (e.g. $N=K^{100}$ becomes $10 \log^{1/2} K$ in the regret)

however

All known contextual bandit algorithms explicitly “keep track” of the $N$ policies. Even worse, just reading in the $N$ would take too long for large $N$. 
“Competing” with a large (even exponentially large) set of policies is commonplace in supervised learning.

- **Targets**: e.g. linear thresholds, CNF, decision trees (in practice only)
- **Methods**: e.g. boosting, SVM, neural networks, gradient descent

The recommendations of the policies don’t need to be explicitly read in when the policy class has structure!

---

Idea originates with [Langford-Zhang '07]
Idea: Use Supervised Learning

- “Competing” with a large (even exponentially large) set of policies is commonplace in supervised learning.
  - **Targets**: e.g. linear thresholds, CNF, decision trees (in practice only)
  - **Methods**: e.g. boosting, SVM, neural networks, gradient descent

- The recommendations of the policies don’t need to be explicitly read in when the policy class has structure!

---

**Warning:** NP-Hard in General

A good policy in $\Pi$

Idea originates with [Langford-Zhang ’07]
Back to Contextual Bandits

context: \( x_1 \) \( x_2 \) \( x_3 \) \( \ldots \) \( x_T \)

1 \( \rightarrow \) $0.70

2 \( \rightarrow \) $0.50

\( \vdots \)

k \( \rightarrow \) 

N experts/policies/functions think of \( N \gg K \)
Back to Contextual Bandits

made-up data

Supervised Learning Oracle

context: $x_1 \ x_2 \ x_3 \ \cdots \ T$

N experts/policies/functions think of $N \gg K$

Supervised Learning Oracle made-up data

1 1
2 5
3 1
$k$ 3
Back to Contextual Bandits

N experts/policies/functions think of $N \gg K$

context: $x_1$, $x_2$, $x_3$, ..., $x_T$

Supervised Learning Oracle

made-up data
Main Theorem [Dudik-Hsu-Kale-Karampatziakis-Langford-R-Zhang ’11]:
For any $\delta > 0$, w.p. at least $1-\delta$, given access to a supervised learning oracle, Randomized-UCB has regret at most $O((KT \ln (NT/\delta))^{1/2} + K \ln(NK/\delta))$ in the stochastic contextual bandit setting and runs in time $\text{poly}(K,T,\ln N)$. 
Randomized-UCB

Main Theorem [Dudik-Hsu-Kale-Karampatziakis-Langford-R-Zhang ’11]:
For any $\delta > 0$, w.p. at least $1 - \delta$, given access to a supervised learning oracle, Randomized-UCB has regret at most $O((KT \ln (NT/\delta))^{1/2} + K \ln(NK/\delta))$ in the stochastic contextual bandit setting and runs in time $\text{poly}(K, T, \ln N)$.

if arms are chosen among only good policies s.t. all have variance < approx 2K, we win

\[ \downarrow \]
\[ \text{can prove this exists via a minimax theorem} \]

\[ \downarrow \]
\[ \text{this condition can be softened to occasionally allow choosing of bad policies} \]
\[ \downarrow \]
\[ \text{via “randomized” upper confidence bounds} \]

\[ \downarrow \]
\[ \text{creates a problem of how to choose arms as to satisfy the constraints} \]
\[ \downarrow \]
\[ \text{expressed as convex optimization problem} \]

\[ \downarrow \]
\[ \text{solvable by ellipsoid algorithm} \]
\[ \downarrow \]
\[ \text{can implement a separation oracle with the supervised learning oracle} \]
Main Theorem [Dudik-Hsu-Kale-Karampatziakis-Langford-R-Zhang ’11]: For any $\delta > 0$, w.p. at least $1 - \delta$, given access to a supervised learning oracle, Randomized-UCB has regret at most $O((KT \ln (NT/\delta))^{1/2} + K \ln(NK/\delta))$ in the stochastic contextual bandit setting and runs in time $\text{poly}(K, T, \ln N)$.

if arms are chosen among only good policies s.t. all have variance $< \text{approx } 2K$, we win

can prove this exists via a minimax theorem
	his condition can be softened to occasionally allow choosing of bad policies via “randomized upper confidence bounds

can implement a separation oracle with the supervised learning oracle

can implement a separation oracle with the supervised learning oracle

Not practical to implement! (yet)
Outline

- The setting and some background
- Show ideas that fail
- Give a high probability optimal algorithm
- Dealing with VC sets
- An efficient algorithm
- Slates
Bandit Slate Problems
[Kale-R-Schapire ’11]

**Problem**: Instead of selecting one arm, we need to select $s \geq 1$, arms (possibly ranked). The motivation is web ads where a search engine shows multiple ads at once.
Slates Setting

- On round $t$ algorithm selects a state $S_t$ of $s$ arms
  - **Unordered** or Ordered
  - **No context** or Contextual

- Algorithm sees $r_j(t)$ for all $j$ in $S$.

- Algorithm gets reward $\sum_{j \in S} r_j(t)$

- Obvious solution is to reduce to the regular bandit problem, but we can do much better.
Algorithm Idea
Algorithm Idea
Algorithm Idea
Algorithm Idea

Multiplicative update

$$\hat{p}_i(t + 1) = p_i(t) \exp(-\eta \ell_i(t)) / Z(t)$$
Algorithm Idea

Relative entropy projection

$$RE(p \mid q) = \sum_i p_i \ln \left( \frac{p_i}{q_i} \right).$$

Also "Component Hedge," independently by Koolen et al. '10.
Algorithm Idea
### Slate Results

<table>
<thead>
<tr>
<th></th>
<th>Unordered Slates</th>
<th>Ordered, with Positional Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Policies</td>
<td>$\tilde{O}(sKT^{1/2})$ *</td>
<td>$\tilde{O}(s(KT)^{1/2})$</td>
</tr>
<tr>
<td>N Policies</td>
<td>$\tilde{O}(sKT \ln N)^{1/2}$</td>
<td>$\tilde{O}(s(KT \ln N)^{1/2})$</td>
</tr>
</tbody>
</table>

*Independently obtained by Uchiya et al. ’10, using different methods.*
The contextual bandit setting captures many interesting real-world problems.

We presented the first optimal, high-probability, contextual algorithm.

We showed how one could possibly make it efficient.
  - Not fully there yet…

We discussed slates – a more real-world setting.
  - How to make those efficient?