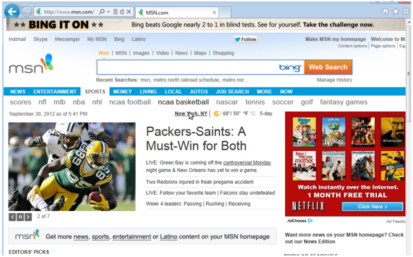
Learning to Explore

John Langford
Microsoft Research

Workshop on Big Data and Statistical Machine Learning, January 26, 2014

git clone git://github.com/JohnLangford/vowpal_wabbit.git

Examples of Interactive Learning



Repeatedly:

1. A user comes to Microsoft (with history of previous visits, IP address, data related to an account)
2. Microsoft chooses information to present (urls, ads, news stories)
3. The user reacts to the presented information (clicks on something, clicks, comes back and clicks again,...)

How do you choose content?

Another Example: Clinical Decision Making

“Whoa—way too much information.”

Repeatedly:

1. A patient comes to a doctor with symptoms, medical history, test results
2. The doctor chooses a treatment
3. The patient responds to it

The doctor wants a policy for choosing targeted treatments for individual patients.

The Contextual Bandit Setting

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

1. The world produces some context $x \in X$
2. The learner chooses an action $a \in A$
3. The world reacts with reward $r_a \in [0, 1]$

Goal: Learn a good policy for choosing actions given context.

The "Direct method"

Use past data to learn a reward predictor $\hat{r}(x, a)$, and act according to $\arg\max_a \hat{r}(x, a)$.

The "Direct method"

Use past data to learn a reward predictor $\hat{r}(x, a)$, and act according to $\arg\max_a \hat{r}(x, a)$.

Example: Deployed policy always takes a_1 on x_1 and a_2 on x_2.

	a_1	a_2
x_1		
x_2		

The "Direct method"

Use past data to learn a reward predictor $\hat{r}(x, a)$, and act according to $\arg\max_a \hat{r}(x, a)$.

Example: Deployed policy always takes a_1 on x_1 and a_2 on x_2.

Observed

	a_1	a_2
x_1	.8	?
x_2	?	.2

The "Direct method"

Use past data to learn a reward predictor $\hat{r}(x, a)$, and act according to $\arg\max_a \hat{r}(x, a)$.

Example: Deployed policy always takes a_1 on x_1 and a_2 on x_2.

Observed/Estimated

	a_1	a_2
x_1	.8/.8	?/.5
x_2	?/.5	.2 /.2

The "Direct method"

Use past data to learn a reward predictor $\hat{r}(x, a)$, and act according to $\arg\max_a \hat{r}(x, a)$.

Example: Deployed policy always takes a_1 on x_1 and a_2 on x_2.

Observed/Estimated

	a_1	a_2
x_1	.8/.8	?/.5
x_2	.3/.5	.2 /.2

The "Direct method"

Use past data to learn a reward predictor $\hat{r}(x, a)$, and act according to $\arg\max_a \hat{r}(x, a)$.

Example: Deployed policy always takes a_1 on x_1 and a_2 on x_2.

Observed/Estimated

	a_1	a_2
x_1	.8/.8	?/.5
x_2	.3/.3	.2 /.2

The "Direct method"

Use past data to learn a reward predictor $\hat{r}(x, a)$, and act according to $\arg\max_a \hat{r}(x, a)$.

Example: Deployed policy always takes a_1 on x_1 and a_2 on x_2.

Observed/Estimated/True

	a_1	a_2
x_1	.8/.8/.8	?/.5/1
x_2	.3/.3/.3	.2 /.2 /.2

The "Direct method"

Use past data to learn a reward predictor $\hat{r}(x, a)$, and act according to $\arg\max_a \hat{r}(x, a)$.

Example: Deployed policy always takes a_1 on x_1 and a_2 on x_2.

Observed/Estimated/True

	a_1	a_2
x_1	.8/.8/.8	?/.5/1
x_2	.3/.3/.3	.2 /.2 /.2

Basic observation 1: Generalization insufficient.

The "Direct method"

Use past data to learn a reward predictor $\hat{r}(x, a)$, and act according to $\arg\max_a \hat{r}(x, a)$.

Example: Deployed policy always takes a_1 on x_1 and a_2 on x_2.

Observed/Estimated/True

	a_1	a_2
x_1	.8/.8/.8	?/.5/1
x_2	.3/.3/.3	.2 /.2 /.2

Basic observation 2: Exploration required.

The "Direct method"

Use past data to learn a reward predictor $\hat{r}(x, a)$, and act according to $\arg\max_a \hat{r}(x, a)$.

Example: Deployed policy always takes a_1 on x_1 and a_2 on x_2.

Observed/Estimated/True

	a_1	a_2
x_1	.8/.8/.8	?/.5/1
x_2	.3/.3/.3	.2 /.2 /.2

Basic observation 3: Errors $\neq$ exploration.

The Evaluation Problem

Let $\pi : X \to A$ be a policy mapping features to actions. How do we evaluate it?

The Evaluation Problem

Let $\pi : X \rightarrow A$ be a policy mapping features to actions. How do we evaluate it?

Method 1: Deploy algorithm in the world.

Very Expensive!

The Importance Weighting Trick

Let $\pi : X \to A$ be a policy mapping features to actions. How do we evaluate it?

The Importance Weighting Trick

Let $\pi : X \to A$ be a policy mapping features to actions. How do we evaluate it?

One answer: Collect T exploration samples

$$(x, a, r_a, p_a),$$

where

$x =$ context

$a =$ action

$r_a =$ reward for action

$p_a =$ probability of action a

then evaluate:

$$\text{Value}(\pi) = \text{ Average}\left(\frac{r_a \mathbf{1}(\pi(x) = a)}{p_a}\right)$$

The Importance Weighting Trick

Theorem

For all policies π, for all IID data distributions D, Value(π) is an unbiased estimate of the expected reward of π:

$$\mathbf{E}_{(x,\vec{r})\sim D}\left[r_{\pi(x)}\right] = \mathbf{E}[\,\text{Value}(\pi)\,]$$

The Importance Weighting Trick

Theorem

For all policies π, for all IID data distributions D, Value(π) is an unbiased estimate of the expected reward of π:

$$\mathbf{E}_{(x,\vec{r})\sim D}\left[r_{\pi(x)}\right] = \mathbf{E}[\,\text{Value}(\pi)\,]$$

Proof: $\mathbf{E}_{a\sim p}\left[\frac{r_a \mathbf{1}(\pi(x)=a)}{p_a}\right] = \sum_a p_a \frac{r_a \mathbf{1}(\pi(x)=a)}{p_a} = r_{\pi(x)}$

The Importance Weighting Trick

Theorem

For all policies π, for all IID data distributions D, Value(π) is an unbiased estimate of the expected reward of π:

$$\mathbf{E}_{(x,\vec{r})\sim D}\left[r_{\pi(x)}\right] = \mathbf{E}[\,\text{Value}(\pi)\,]$$

Proof: $\mathbf{E}_{a\sim p}\left[\frac{r_a \mathbf{1}(\pi(x)=a)}{p_a}\right] = \sum_a p_a \frac{r_a \mathbf{1}(\pi(x)=a)}{p_a} = r_{\pi(x)}$

Example:

Action	1	2
Reward	0.5	1
Probability	$\frac{1}{4}$	$\frac{3}{4}$
Estimate	2 | 0	0 | $\frac{4}{3}$

How do you test things?

Use format:

action:cost:probability | features

Example:

1:1:0.5 | tuesday year million short compan vehicl line stat financ commit exchang plan corp subsid credit issu debt pay gold bureau prelimin refin billion telephon time draw basic relat file spokesm reut secur acquir form prospect period interview regist toront resourc barrick ontario qualif bln prospectus convertibl vinc borg arequip

...

How do you train?

How do you train?

Reduce to cost-sensitive classification: $(x, \vec{c})$

How do you train?

Reduce to cost-sensitive classification: $(x, \vec{c})$

vw –cb 2 –cb_type dr rcv1.train.txt.gz -c –ngram 2 –skips 4 -b 24 -l 0.25

Progressive 0/1 loss: 0.04582

vw –cb 2 –cb_type ips rcv1.train.txt.gz -c –ngram 2 –skips 4 -b 24 -l 0.125

Progressive 0/1 loss: 0.05065

vw –cb 2 –cb_type dm rcv1.train.txt.gz -c –ngram 2 –skips 4 -b 24 -l 0.125

Progressive 0/1 loss: 0.04679

Reminder: Contextual Bandit Setting

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

1. The world produces some context $x \in X$
2. The learner chooses an action $a \in A$
3. The world reacts with reward $r_a \in [0, 1]$

Goal: Learn a good policy for choosing actions given context.

What does learning mean? Efficiently competing with some large reference class of policies $\Pi = \{\pi : X \to A\}$:

$$\text{Regret} = \max_{\pi \in \Pi} \text{average}_t (r_{\pi(x)} - r_a)$$

Building an Algorithm

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

1. The world produces some context $x \in X$

3. The learner chooses an action $a \in A$
4. The world reacts with reward $r_a \in [0, 1]$

Building an Algorithm

Let Q_1 = uniform distribution
For $t = 1, \ldots, T$:

1. The world produces some context $x \in X$
2. Draw $\pi \sim Q_t$
3. The learner chooses an action $a \in A$ using $\pi(x)$.
4. The world reacts with reward $r_a \in [0, 1]$
5. Update Q_{t+1}

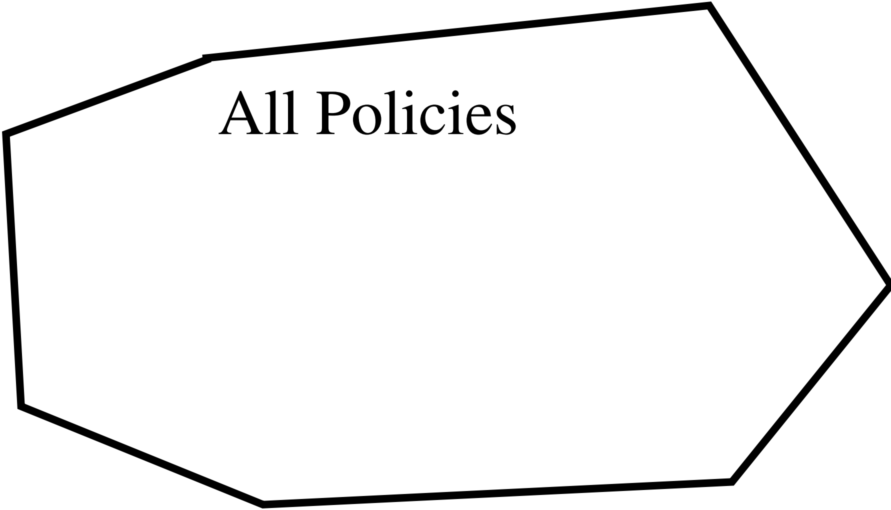
What is good Q_t?

- **Exploration:** Q_t allows discovery of good policies
- **Exploitation:** Q_t large on good policies

How do you find Q_t?

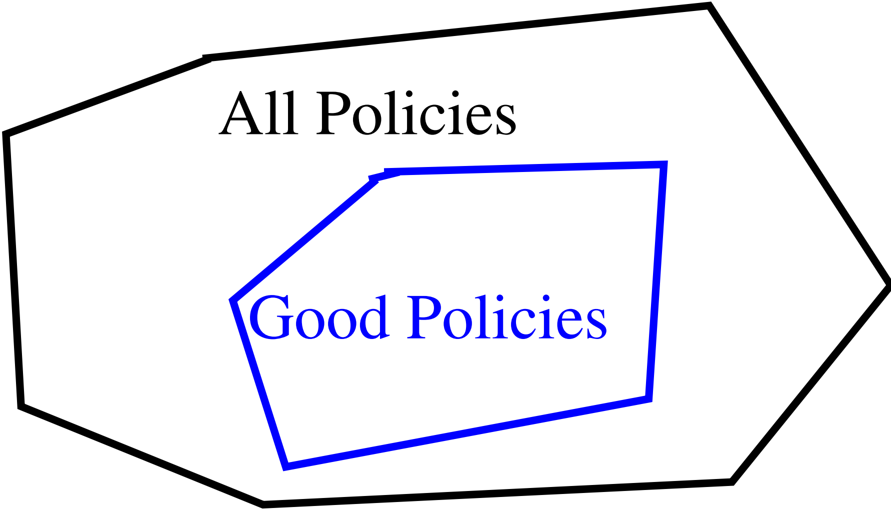
How do you find Q_t?

crudely: by creating a cover.



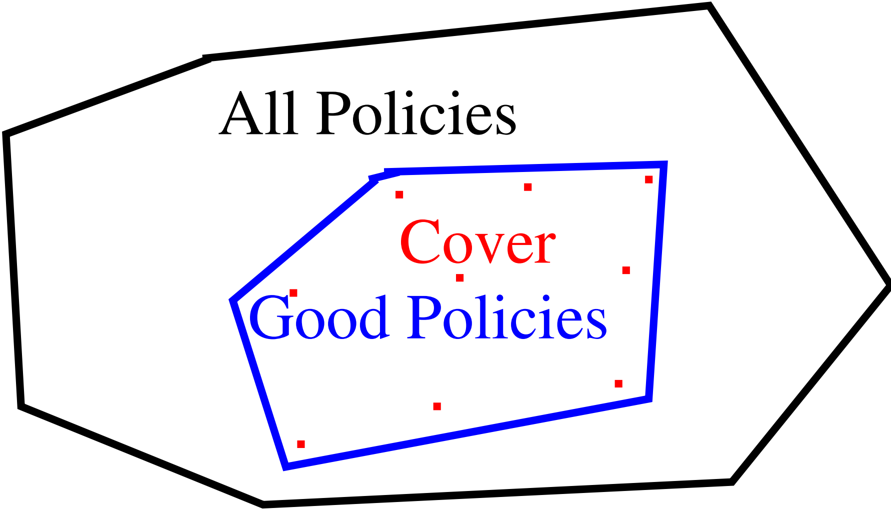
How do you find Q_t?

crudely: by creating a cover.



How do you find Q_t?

crudely: by creating a cover.



How do you create a cover?

How do you create a cover?

Reduce to Cost sensitive classification: $(x, \vec{c})$

How do you create a cover?

Reduce to Cost sensitive classification: $(x, \vec{c})$

Let $\mu = \frac{1}{\sqrt{T|A|}}$ = minimum probability.

How do you create a cover?

Reduce to Cost sensitive classification: $(x, \vec{c})$

Let $\mu = \frac{1}{\sqrt{T|A|}}$ = minimum probability.

Let $Q_n(a) = \max\{\mu, \frac{1}{n}\sum_i I(\pi_i(x) = a)\}$

How do you create a cover?

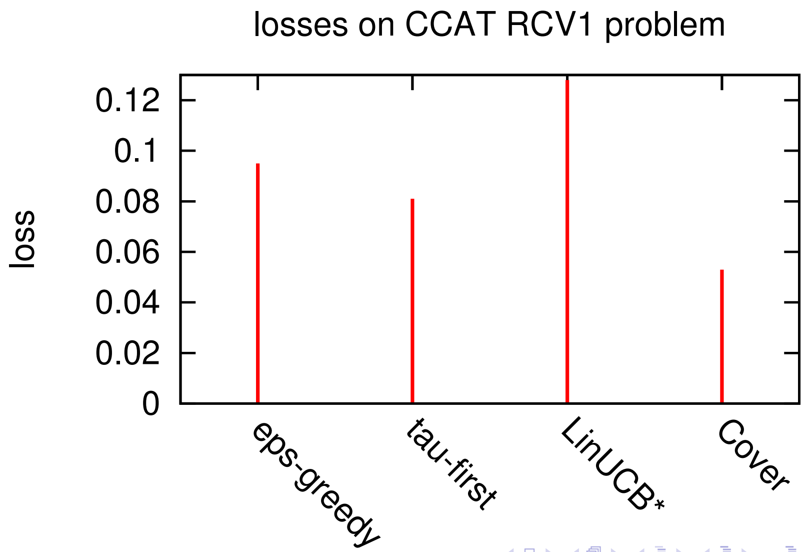
Reduce to Cost sensitive classification: $(x, \vec{c})$

Let $\mu = \frac{1}{\sqrt{T|A|}}$ = minimum probability.

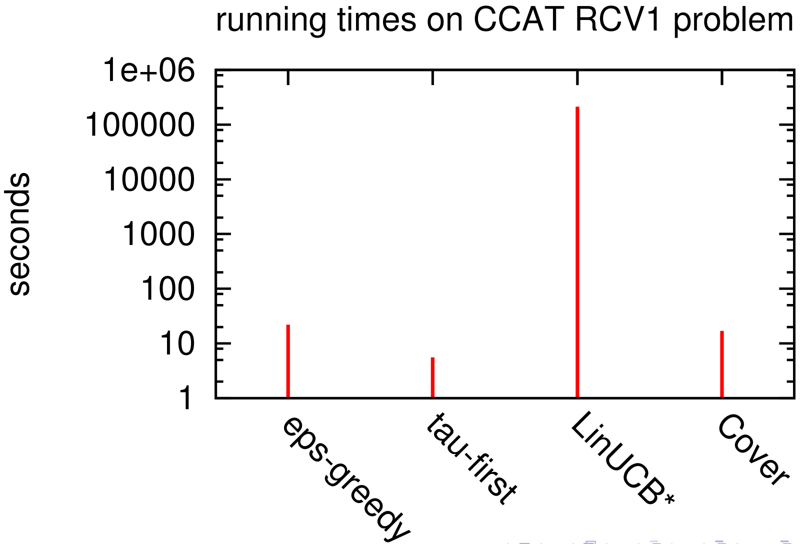
Let $Q_n(a) = \max\{\mu, \frac{1}{n}\sum_i I(\pi_i(x) = a)\}$

$c_a =$ unbiased cost estimate $- \epsilon \frac{\mu}{Q_n(a)}$

How well does this work?



How long does this take?



Further reading

VW wiki: https://github.com/JohnLangford/vowpal_wabbit/wiki
NIPS tutorial: http://hunch.net/~jl/interact.pdf

Code Vowpal Wabbit open source project, http://github.com/JohnLangford/vowpal_wabbit/wiki, 2007.

Explore A. Agarwal, D. Hsu, S. Kale, J. Langford, L. Li, R. Schapire, Taming the Monster: A Fast and Simple Algorithm for Contextual Bandits, http://arxiv.org/abs/1402.0555