Microsoft Research

Learning to Interact

John Langford @ Microsoft Research (with help from many)

Slides at: http://hunch.net/~jl/interact.pdf

For demo:
Raw RCV1 CCAT-or-not:
http://hunch.net/~jl/VW_raw.tar.gz
Simple converter: wget http://hunch.net/~jl/cbify.cc
Vowpal Wabbit for learning: http://hunch.net/~vw

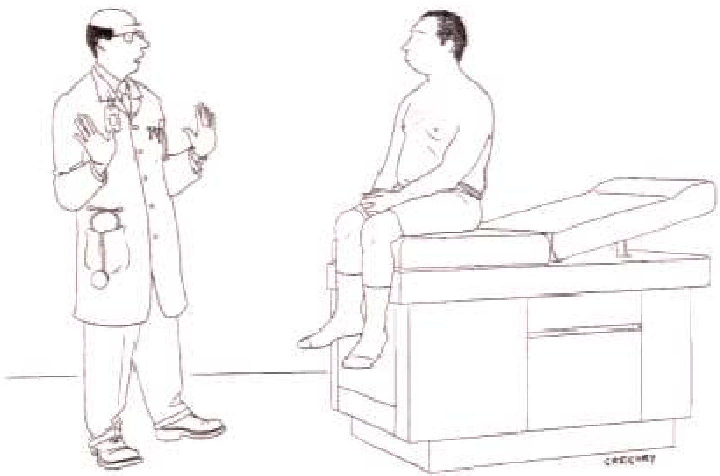
Examples of Interactive Learning

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,...)

Microsoft wants to interactively choose content and use the observed feedback to improve future content choices.

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.

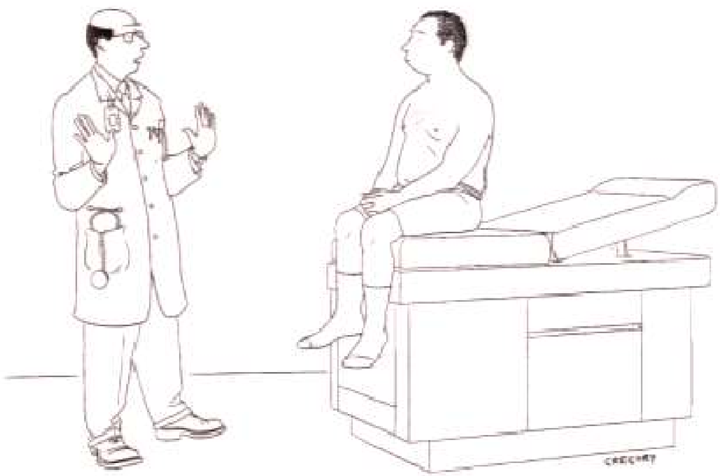
Examples of Interactive Learning

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,...)

Microsoft wants to interactively choose content and use the observed feedback to improve future content choices.

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 Evaluation Problem

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

Method 1: Deploy algorithm in the world.

Very Expensive!

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 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/True

	a_1	a_2
x_1	.8/.8/.8	?/.514/1
x_2	.3/.3/.3	.2 /.014 /.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	?/.514/1
x_2	.3/.3/.3	.2 /.014 /.2

Basic observation 1: Generalization alone is not sufficient.

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	?/.514/1
x_2	.3/.3/.3	.2 /.014 /.2

Basic observation 3: Prediction errors not controlled exploration.

Outline

1. Using Exploration
   1. Problem Definition
   2. Direct Method fails
   3. Importance Weighting
   4. Missing Probabilities
   5. Doubly Robust
2. Doing Exploration

Method 3: The Importance Weighting Trick

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

Method 3: 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 of the form

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

where
x = context
a = action
r_a = reward for action
p_a = probability of action a
then evaluate:

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

Method 3: 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 of the form

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

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:

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

with deviations bounded by

$$O\left(\frac{1}{\sqrt{T \min_x p_{\pi(x)}}}\right)$$

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

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:

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

with deviations bounded by

$$O\left(\frac{1}{\sqrt{T \min_x p_{\pi(x)}}}\right)$$

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

What if you don't know probabilities?

Suppose p was:

1. misrecorded "We randomized some actions, but then the Business Logic did something else."
2. not recorded "We randomized some scores which had an unclear impact on actions".
3. nonexistent "On Tuesday we did A and on Wednesday B".

What if you don't know probabilities?

Suppose p was:

1. misrecorded "We randomized some actions, but then the Business Logic did something else."
2. not recorded "We randomized some scores which had an unclear impact on actions".
3. nonexistent "On Tuesday we did A and on Wednesday B".

Learn predictor $\hat{p}(a|x)$ on $(x, a)^*$ data.

Define new estimator: $\hat{V}(\pi) = \hat{E}_{x,a,r_a}\left[\frac{r_a I(\pi(x)=a)}{\max\{\tau, \hat{p}(a|x)\}}\right]$ where $\tau =$ small number.

What if you don't know probabilities?

Suppose p was:

1. misrecorded "We randomized some actions, but then the Business Logic did something else."
2. not recorded "We randomized some scores which had an unclear impact on actions".
3. nonexistent "On Tuesday we did A and on Wednesday B".

Learn predictor $\hat{p}(a|x)$ on $(x, a)^*$ data.

Define new estimator: $\hat{V}(\pi) = \hat{E}_{x,a,r_a}\left[\frac{r_a I(\pi(x)=a)}{\max\{\tau,\hat{p}(a|x)\}}\right]$ where $\tau =$ small number.

Theorem: For all IID D, for all policies π with $p(a|x) > \tau$

$$|\text{Value}(\pi) - E\hat{V}(\pi)| \leq \frac{\sqrt{\text{reg}(\hat{p})}}{\tau}$$

where $\text{reg}(\hat{p}) = \mathbf{E}_{x\sim D, a\sim p(a|x)}[(p(a|x) - \hat{p}(a|x))^2]$ = squared loss regret.

What if you don't know probabilities?

Suppose p was:

1. misrecorded "We randomized some actions, but then the Business Logic did something else."
2. not recorded "We randomized some scores which had an unclear impact on actions".
3. nonexistent "On Tuesday we did A and on Wednesday B".

Learn predictor $\hat{p}(a|x)$ on $(x, a)^*$ data.

Define new estimator: $\hat{V}(\pi) = \hat{E}_{x,a,r_a}\left[\frac{r_a I(\pi(x)=a)}{\max\{\tau, \hat{p}(a|x)\}}\right]$ where $\tau =$ small number.

What if you don't know probabilities?

Suppose p was:

1. misrecorded "We randomized some actions, but then the Business Logic did something else."
2. not recorded "We randomized some scores which had an unclear impact on actions".
3. nonexistent "On Tuesday we did A and on Wednesday B".

Learn predictor $\hat{p}(a|x)$ on $(x, a)^*$ data.

Define new estimator: $\hat{V}(\pi) = \hat{E}_{x,a,r_a}\left[\frac{r_a I(\pi(x)=a)}{\max\{\tau, \hat{p}(a|x)\}}\right]$ where $\tau =$ small number.

Theorem: For all IID D, for all policies π with $p(a|x) > \tau$

$$|\text{Value}(\pi) - E\hat{V}(\pi)| \leq \frac{\sqrt{\text{reg}(\hat{p})}}{\tau}$$

where $\text{reg}(\hat{p}) = \mathbf{E}_{x \sim D, a \sim p(a|x)}[(p(a|x) - \hat{p}(a|x))^2]$ = squared loss regret.

What if you don't know probabilities?

Suppose p was:

1. misrecorded "We randomized some actions, but then the Business Logic did something else."
2. not recorded "We randomized some scores which had an unclear impact on actions".
3. nonexistent "On Tuesday we did A and on Wednesday B".

Method 3: 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 of the form

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

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:

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

with deviations bounded by

$$O\left(\frac{1}{\sqrt{T \min_x p_{\pi(x)}}}\right)$$

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

What if you don't know probabilities?

Suppose p was:

1. misrecorded "We randomized some actions, but then the Business Logic did something else."
2. not recorded "We randomized some scores which had an unclear impact on actions".
3. nonexistent "On Tuesday we did A and on Wednesday B".

What if you don't know probabilities?

Suppose p was:

1. misrecorded "We randomized some actions, but then the Business Logic did something else."
2. not recorded "We randomized some scores which had an unclear impact on actions".
3. nonexistent "On Tuesday we did A and on Wednesday B".

Learn predictor $\hat{p}(a|x)$ on $(x, a)^*$ data.
Define new estimator: $\hat{V}(\pi) = \hat{E}_{x,a,r_a}\left[\frac{r_a I(\pi(x)=a)}{\max\{\tau,\hat{p}(a|x)\}}\right]$ where $\tau =$ small number.

Theorem: For all IID D, for all policies π with $p(a|x) > \tau$

$$|\text{Value}(\pi) - E\hat{V}(\pi)| \leq \frac{\sqrt{\text{reg}(\hat{p})}}{\tau}$$

where $\text{reg}(\hat{p}) = \mathbf{E}_{x\sim D, a\sim p(a|x)}[(p(a|x) - \hat{p}(a|x))^2]$ = squared loss regret.

Can we do better?

Suppose we have a (possibly bad) reward estimator $\hat{r}(a, x)$. How can we use it?

Can we do better?

Suppose we have a (possibly bad) reward estimator $\hat{r}(a, x)$. How can we use it?

$$\text{Value}'(\pi) = \text{Average}\left(\frac{(r_a - \hat{r}(a, x))\mathbf{1}(\pi(x) = a)}{p_a} + \hat{r}(\pi(x), x)\right)$$

Can we do better?

Suppose we have a (possibly bad) reward estimator $\hat{r}(a,x)$. How can we use it?

$$\text{Value}'(\pi) = \text{Average}\left(\frac{(r_a - \hat{r}(a,x))\mathbf{1}(\pi(x)=a)}{p_a} + \hat{r}(\pi(x),x)\right)$$

Let $\Delta(a,x) = \hat{r}(a,x) - E_{\vec{r}|x} r_a$ = reward deviation
Let $\delta(a,x) = 1 - \frac{p_a}{\hat{p}_a}$ = probability deviation

Theorem

For all policies π and all $(x,\vec{r})$:

$$|\text{Value}'(\pi) - E_{\vec{r}|x}[r_{\pi(x)}]| \leq |\Delta(\pi(x),x)\delta(\pi(x),x)|$$

The deviations multiply, so deviations < 1 means we win!

How do you test things?

Contextual Bandit datasets tend to be highly proprietary. What can you do?

How do you test things?

Contextual Bandit datasets tend to be highly proprietary. What can you do?

1. Pick classification dataset.
2. Generate (x, a, r, p) quads via uniform random exploration of actions

Apply transform to RCV1 dataset.

wget `http://hunch.net/~jl/VW_raw.tar.gz`

wget `http://hunch.net/~jl/cbify.cc`

Output format is:

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?

1. Learn $\hat{r}(a, x)$.
2. Compute for each x the double-robust estimate for each $a' \in \{1, ..., K\}$:

$$\frac{(r - \hat{r}(a, x))I(a' = a)}{p(a|x)} + \hat{r}(a', x)$$

3. Learn π using a cost-sensitive classifier. We'll use Vowpal Wabbit: `http://hunch.net/~vw`

How do you test things?

Contextual Bandit datasets tend to be highly proprietary. What can you do?

1. Pick classification dataset.
2. Generate (x, a, r, p) quads via uniform random exploration of actions

Apply transform to RCV1 dataset.

wget `http://hunch.net/~jl/VW_raw.tar.gz`

wget `http://hunch.net/~jl/cbify.cc`

Output format is:

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?

1. Learn $\hat{r}(a, x)$.
2. Compute for each x the double-robust estimate for each $a' \in \{1, ..., K\}$:

$$\frac{(r - \hat{r}(a, x))I(a' = a)}{p(a|x)} + \hat{r}(a', x)$$

3. Learn π using a cost-sensitive classifier. We'll use Vowpal Wabbit: `http://hunch.net/~vw`

How do you train?

1. Learn $\hat{r}(a, x)$.
2. Compute for each x the double-robust estimate for each $a' \in \{1, ..., K\}$:

$$\frac{(r - \hat{r}(a, x))I(a' = a)}{p(a|x)} + \hat{r}(a', x)$$

3. Learn π using a cost-sensitive classifier. We'll use Vowpal Wabbit: http://hunch.net/~vw

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

Terminal   4:25   7:06 PM John Langford

```
initial_t = 0
power_t = 0.5
using cache_file = rcv1_train.cb_vw.cache
ignoring text input in favor of cache input
num sources = 1
average    since         example     example  current  current  current
loss       last          counter      weight    label  predict features
*estimate* *estimate*                                                 avglossreg last pred  last correc
t
0.666667   0.666667            3         3.0    known        2      316   0.334247   0.041716   0.000000
0.333333   0.000000            6         6.0    known        2      160   0.328435   0.016708   1.000000
0.365390   0.403858           11        11.0    known        2      202   0.354719   0.040916   0.000000
0.363327   0.361265           22        22.0    known        2      502   0.344410   0.049526   0.000000
0.370952   0.378576           44        44.0    known        2      370   0.405983   0.078159   0.000000
0.288965   0.205072           87        87.0    known        1      340   0.356304   0.100344   1.000000
0.293865   0.298764          174       174.0    known        2      130   0.322963   0.083125   0.000000
0.198690   0.103516          348       348.0    known        2      262   0.297750   0.357253   1.000000
0.158162   0.117633          696       696.0    known        2      124   0.249183   0.082325   0.000000
0.123245   0.088328         1392      1392.0    known        2     1066   0.215804   0.583740   0.000000
0.111740   0.100234         2784      2784.0    known        1      280   0.176151   0.247207   1.000000
0.092496   0.073252         5568      5568.0    known        1      514   0.143719   0.203254   0.000000
0.082852   0.073207        11135     11135.0    known        2      352   0.121448   1.058181   1.000000
0.072335   0.061816        22269     22269.0    known        2      820   0.101361   0.076899   0.000000
0.064118   0.055902        44537     44537.0    known        2      226   0.086304  -0.138273   0.000000
0.059023   0.053927        89073     89073.0    known        1      142   0.074598   1.061901   1.000000
0.054813   0.050603       178146    178146.0    known        2      274   0.065937   1.007291   1.000000
0.050256   0.045699       356291    356291.0    known        1      580   0.059258   1.076878   1.000000
0.046211   0.042166       712582    712582.0    known        1      394   0.053942   0.008066   0.000000

finished run
number of examples = 781265
weighted example sum = 7.813e+05
weighted label sum = 0
average loss = 0.04582
best constant = 0
total feature number = 343993166
 6:15PM 1-of-3-7:                                              ~/presentations/nips_2013 [jl/ttypts/0]
```

Terminal 4:25 7:06 PM John Langford

6:15PM 1-of-3-7: l ~/presentations/nips_2013 [jl/ttypts/0]

Terminal 4:13 7:06 PM John Langford

```
1:1:0.5 | tuesday year million short compan vehicl line stat financ commit exchang plan corp subsid cr
edit 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 prospect
us convertibl vinc borg arequip
1:0:0.5 | econom stock rate month year invest week produc report govern pric index million shar end re
serv foreign research inflat gdp growth export consum output annual industr cent exchang project trad
fisc servic base compar prev money bank debt balanc gold daily import agricultur ago estimat ton preli
min deficit currenc nation call march survey account offic sourc council silf data key apply aug incom
 real indian wholesal current net m3 monitor cumulat bombay bse india extern kg sep jul jun apr indica
t capit ratio pct refer yr weekend bln fii rupee rbi populat forex int sdr delh foodgrain rs gm nca cm
ie wp
1:0:0.5 | tuesday month year govern foreign put gener countr schedul unit stat commit plan includ forc
 chief cancel joint need issu held polic hold congress britain time call host sourc told extent activ
review agree hear arm soldy advis study involut offent malays singapor territ manil duty philippin mil
it conduc clarif train exercis jurisdict crim crimin
1:0:0.5 | bring year ahead point decemb free report strong govern million drop rais short return move
countr april name europ support make presid region trad group early forc revital part commun july memb
 joint prepar join cultur issu monday respect daily freedom bid wednesday sign press call won relat of
fic left list defend damag effort hope remov janu told propos ask secur enter union agree lead discuss
 visit pass confer belief eu european deput tour trip ecolog contact give pact follow resolut western
right parlia coop islam leav organ tie accord moslem turkey nato nuclear kuwait ali turk initiat bulga
r rout restor mutual drug competit roman greec morocc sofia crim bulg balk counterpart ambassador exod
us ethnic era traffick relig standart combat emil constantinescu emirat overthrow mistreat oic petar t
odor zhivkov interced stoyanov suleym demirel
1:0:0.5 | begin free week report strong million end city led countr start unit peopl stat decid pow ce
nt plan make presid forc blam take post campaign monday weak estimat problem wednesday moderat fear bi
g nation provid direct thing caus account fight captur offic defend remov key face push told warn judg
 ask proceed respons separat war troop accus washington failur hand arm origin octob suggest act belie
f prim minist bord particip stick interview town civil western fled southern individual attack map cop
 capit blow paul possibl mine insid deny milit army allegat consequ deflect order thought train commen
d atroc lash defeat eastern rebel refug diamons camp wouldn tuts genoc hutus zair rwand hutu repris ug
and revolt zairean massacr cong hat overthrow keng kagam mobutu sese kigal monst crossroad kinshas new
pap
1:1:0.5 | market invest week fell million end compan industr increas rose total trad group tax money r
evis retail billion nation march thursday institut fund asset prior exempt mutual taxabl ici
1:0:0.5 | clos year decemb remain head fact meet commit intern execut presid operat sale cut july flag
rcv1_train.cb_vw 3192/375458083 bytes (0%)
```


Terminal
7:07 PM
John Langford
Terminal
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
1:0:0.5 | econom stock rate month year invest week product report govern pric index million shar end reserv foreign research inflat gdp growth export consum output annual industr cent exchang project trad fisc servic base compar prev money bank debt balanc gold daily import agricultur ago estimat ton prelimin deficit currenc nation call march survey account offic sourc council silf data key apply aug incom real indian wholesal current net m3 mon
rcv1_train.cb_vw 730/375458083 bytes (0%)
Adobe Reader 9
diff
diff.txt
ICML2012_22Sept.numbers
patch
preferred-web-browser.desktop
Reviewer%20Subject%20Areas.xlsx
wordpress_wp_20061110_124.sql.gz
256.sql.gz
Skype.lnk
Skype
figures
windows
chelseapiers.pdf
Icewind Dale II
Scanned Document.pdf

Terminal 4:13 7:07 PM John Langford

```
1:1:0.5 | tuesday year million short compan vehicl line stat financ commit exchang plan corp subsid cr
edit 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 prospect
us convertibl vinc borg arequip
1:0:0.5 | econom stock rate month year invest week produc report govern pric index million shar end re
serv foreign research inflat gdp growth export consum output annual industr cent exchang project trad
fisc servic base compar prev money bank debt balanc gold daily import agricultur ago estimat ton preli
min deficit currenc nation call march survey account offic sourc council silf data key apply aug incom
 real indian wholesal current net m3 monitor cumulat bombay bse india extern kg sep jul jun apr indica
t capit ratio pct refer yr weekend bln fii rupee rbi populat forex int sdr delh foodgrain rs gm nca cm
ie wp
1:0:0.5 | tuesday month year govern foreign put gener countr schedul unit stat commit plan includ forc
 chief cancel joint need issu held polic hold congress britain time call host sourc told extent activ
review agree hear arm soldy advis study involut offent malays singapor territ manil duty philippin mil
it conduc clarif train exercis jurisdict crim crimin
1:0:0.5 | bring year ahead point decemb free report strong govern million drop rais short return move
countr april name europ support make presid region trad group early forc revital part commun july memb
 joint prepar join cultur issu monday respect daily freedom bid wednesday sign press call won relat of
fic left list defend damag effort hope remov janu told propos ask secur enter union agree lead discuss
 visit pass confer belief eu european deput tour trip ecolog contact give pact follow resolut western
right parlia coop islam leav organ tie accord moslem turkey nato nuclear kuwait ali turk initiat bulga
r rout restor mutual drug competit roman greec morocc sofia crim bulg balk counterpart ambassador exod
us ethnic era traffick relig standart combat emil constantinescu emirat overthrow mistreat oic petar t
odor zhivkov interced stoyanov suleym demirel
1:0:0.5 | begin free week report strong million end city led countr start unit peopl stat decid pow ce
nt plan make presid forc blam take post campaign monday weak estimat problem wednesday moderat fear bi
g nation provid direct thing caus account fight captur offic defend remov key face push told warn judg
 ask proceed respons separat war troop accus washington failur hand arm origin octob suggest act belie
f prim minist bord particip stick interview town civil western fled southern individual attack map cop
 capit blow paul possibl mine insid deny milit army allegat consequ deflect order thought train commen
d atroc lash defeat eastern rebel refug diamons camp wouldn tuts genoc hutus zair rwand hutu repris ug
and revolt zairean massacr cong hat overthrow keng kagam mobutu sese kigal monst crossroad kinshas new
pap
1:1:0.5 | market invest week fell million end compan industr increas rose total trad group tax money r
evis retail billion nation march thursday institut fund asset prior exempt mutual taxabl ici
1:0:0.5 | clos year decemb remain head fact meet commit intern execut presid operat sale cut july flag
rcv1_train.cb_vw 3192/375458083 bytes (0%)
```

Terminal 4:13 7:07 PM John Langford

```
 6:15PM 1-of-3-7: ls -l rcv1_train.cb_vw                    ~/presentations/nips_2013 [jl/ttypts/0]
-rw-r--r-- 1 jl jl 375458083 Dec  5 13:03 rcv1_train.cb_vw
 7:06PM 1-of-3-8: less rcv1_train.cb_vw                     ~/presentations/nips_2013 [jl/ttypts/0]
 7:07PM 1-of-3-9:                                           ~/presentations/nips_2013 [jl/ttypts/0]
```

```
 6:15PM 1-of-3-7: ls -l rcv1_train.cb_vw                          ~/presentations/nips_2013 [jl/ttypts/0]
-rw-r--r-- 1 jl jl 375458083 Dec  5 13:03 rcv1_train.cb_vw
 7:06PM 1-of-3-8: less rcv1_train.cb_vw                           ~/presentations/nips_2013 [jl/ttypts/0]
 7:07PM 1-of-3-9: vw --cb 2 --cb_type dr --ngram 2 --skips 4 -b 24 -l 0.25 rcv1_train.cb_vw -c
Generating 2-grams for all namespaces.
Generating 4-skips for all namespaces.
Num weight bits = 24
learning rate = 0.25
initial_t = 0
power_t = 0.5
using cache_file = rcv1_train.cb_vw.cache
ignoring text input in favor of cache input
num sources = 1
average    since         example     example  current  current  current
loss       last          counter      weight    label  predict features
*estimate* *estimate*                                                   avglossreg last pred  last correc
t
0.666667   0.666667          3         3.0    known        2       316  0.334247   0.041716   0.000000
0.333333   0.000000          6         6.0    known        2       160  0.328435   0.016708   1.000000
0.365390   0.403858         11        11.0    known        2       202  0.354719   0.040916   0.000000
0.363327   0.361265         22        22.0    known        2       502  0.344410   0.049526   0.000000
0.370952   0.378576         44        44.0    known        2       370  0.405983   0.078159   0.000000
0.288965   0.205072         87        87.0    known        1       340  0.356304   0.100344   1.000000
0.293865   0.298764        174       174.0    known        2       130  0.322963   0.083125   0.000000
0.198690   0.103516        348       348.0    known        2       262  0.297750   0.357253   1.000000
0.158162   0.117633        696       696.0    known        2       124  0.249183   0.082325   0.000000
0.123245   0.088328       1392      1392.0    known        2      1066  0.215804   0.583740   0.000000
0.111740   0.100234       2784      2784.0    known        1       280  0.176151   0.247207   1.000000
```

Terminal 4:16 7:08 PM John Langford

```
 6:15PM 1-of-3-7: ls -l rcv1_train.cb_vw                              ~/presentations/nips_2013 [jl/ttypts/0]
-rw-r--r-- 1 jl jl 375458083 Dec  5 13:03 rcv1_train.cb_vw
 7:06PM 1-of-3-8: less rcv1_train.cb_vw                               ~/presentations/nips_2013 [jl/ttypts/0]
 7:07PM 1-of-3-9: vw --cb 2 --cb_type dr --ngram 2 --skips 4 -b 24 -l 0.25 rcv1_train.cb_vw -c
Generating 2-grams for all namespaces.
Generating 4-skips for all namespaces.
Num weight bits = 24
learning rate = 0.25
initial_t = 0
power_t = 0.5
using cache_file = rcv1_train.cb_vw.cache
ignoring text input in favor of cache input
num sources = 1
average     since           example     example  current  current  current
loss        last            counter      weight    label  predict features
*estimate* *estimate*                                                        avglossreg last pred  last correc
t
0.666667   0.666667          3        3.0    known          2        316   0.334247   0.041716   0.000000
0.333333   0.000000          6        6.0    known          2        160   0.328435   0.016708   1.000000
0.365390   0.403858         11       11.0    known          2        202   0.354719   0.040916   0.000000
0.363327   0.361265         22       22.0    known          2        502   0.344410   0.049526   0.000000
0.370952   0.378576         44       44.0    known          2        370   0.405983   0.078159   0.000000
0.288965   0.205072         87       87.0    known          1        340   0.356304   0.100344   1.000000
0.293865   0.298764        174      174.0    known          2        130   0.322963   0.083125   0.000000
0.198690   0.103516        348      348.0    known          2        262   0.297750   0.357253   1.000000
0.158162   0.117633        696      696.0    known          2        124   0.249183   0.082325   0.000000
0.123245   0.088328       1392     1392.0    known          2       1066   0.215804   0.583740   0.000000
0.111740   0.100234       2784     2784.0    known          1        280   0.176151   0.247207   1.000000
0.092496   0.073252       5568     5568.0    known          1        514   0.143719   0.203254   0.000000
0.082852   0.073207      11135    11135.0    known          2        352   0.121448   1.058181   1.000000
0.072335   0.061816      22269    22269.0    known          2        820   0.101361   0.076899   0.000000
```

```
Terminal                                                                 4:16        7:08 PM  John Langford
 6:15PM 1-of-3-7: ls -l rcv1_train.cb_vw                         ~/presentations/nips_2013 [jl/ttypts/0]
-rw-r--r-- 1 jl jl 375458083 Dec  5 13:03 rcv1_train.cb_vw
 7:06PM 1-of-3-8: less rcv1_train.cb_vw                          ~/presentations/nips_2013 [jl/ttypts/0]
 7:07PM 1-of-3-9: vw --cb 2 --cb_type dr --ngram 2 --skips 4 -b 24 -l 0.25 rcv1_train.cb_vw -c
Generating 2-grams for all namespaces.
Generating 4-skips for all namespaces.
Num weight bits = 24
learning rate = 0.25
initial_t = 0
power_t = 0.5
using cache_file = rcv1_train.cb_vw.cache
ignoring text input in favor of cache input
num sources = 1
average    since         example     example  current  current  current
loss       last          counter      weight    label  predict features
*estimate* *estimate*                                                    avglossreg last pred  last correc
t
0.666667   0.666667            3         3.0    known        2      316  0.334247   0.041716   0.000000
0.333333   0.000000            6         6.0    known        2      160  0.328435   0.016708   1.000000
0.365390   0.403858           11        11.0    known        2      202  0.354719   0.040916   0.000000
0.363327   0.361265           22        22.0    known        2      502  0.344410   0.049526   0.000000
0.370952   0.378576           44        44.0    known        2      370  0.405983   0.078159   0.000000
0.288965   0.205072           87        87.0    known        1      340  0.356304   0.100344   1.000000
0.293865   0.298764          174       174.0    known        2      130  0.322963   0.083125   0.000000
0.198690   0.103516          348       348.0    known        2      262  0.297750   0.357253   1.000000
0.158162   0.117633          696       696.0    known        2      124  0.249183   0.082325   0.000000
0.123245   0.088328         1392      1392.0    known        2     1066  0.215804   0.583740   0.000000
0.111740   0.100234         2784      2784.0    known        1      280  0.176151   0.247207   1.000000
0.092496   0.073252         5568      5568.0    known        1      514  0.143719   0.203254   0.000000
0.082852   0.073207        11135     11135.0    known        2      352  0.121448   1.058181   1.000000
0.072335   0.061816        22269     22269.0    known        2      820  0.101361   0.076899   0.000000
0.064118   0.055902        44537     44537.0    known        2      226  0.086304   -0.138273  0.000000
0.059023   0.053927        89073     89073.0    known        1      142  0.074598   1.061901   1.000000
0.054813   0.050603       178146    178146.0    known        2      274  0.065937   1.007291   1.000000
```

Terminal 3:03 7:09 PM John Langford

```
initial_t = 0
power_t = 0.5
using cache_file = rcv1_train.cb_vw.cache
ignoring text input in favor of cache input
num sources = 1
average    since          example     example  current  current  current
loss       last           counter      weight    label  predict features
*estimate* *estimate*                                                  avglossreg last pred  last correc
t
0.666667   0.666667             3         3.0    known        2      316   0.334247   0.041716   0.000000
0.333333   0.000000             6         6.0    known        2      160   0.328435   0.016708   1.000000
0.365390   0.403858            11        11.0    known        2      202   0.354719   0.040916   0.000000
0.363327   0.361265            22        22.0    known        2      502   0.344410   0.049526   0.000000
0.370952   0.378576            44        44.0    known        2      370   0.405983   0.078159   0.000000
0.288965   0.205072            87        87.0    known        1      340   0.356304   0.100344   1.000000
0.293865   0.298764           174       174.0    known        2      130   0.322963   0.083125   0.000000
0.198690   0.103516           348       348.0    known        2      262   0.297750   0.357253   1.000000
0.158162   0.117633           696       696.0    known        2      124   0.249183   0.082325   0.000000
0.123245   0.088328          1392      1392.0    known        2     1066   0.215804   0.583740   0.000000
0.111740   0.100234          2784      2784.0    known        1      280   0.176151   0.247207   1.000000
0.092496   0.073252          5568      5568.0    known        1      514   0.143719   0.203254   0.000000
0.082852   0.073207         11135     11135.0    known        2      352   0.121448   1.058181   1.000000
0.072335   0.061816         22269     22269.0    known        2      820   0.101361   0.076899   0.000000
0.064118   0.055902         44537     44537.0    known        2      226   0.086304  -0.138273   0.000000
0.059023   0.053927         89073     89073.0    known        1      142   0.074598   1.061901   1.000000
0.054813   0.050603        178146    178146.0    known        2      274   0.065937   1.007291   1.000000
0.050256   0.045699        356291    356291.0    known        1      580   0.059258   1.076878   1.000000
0.046211   0.042166        712582    712582.0    known        1      394   0.053942   0.008066   0.000000

finished run
number of examples = 781265
weighted example sum = 7.813e+05
weighted label sum = 0
average loss = 0.04582
best constant = 0
total feature number = 343993166
 7:09PM 1-of-3-10:                                        ~/presentations/nips_2013 [jl/ttypts/0]
```

Terminal   4:06   7:09 PM   John Langford

```
0.046211   0.042166     712582 712582.0    known         1       394   0.053942   0.008066   0.000000

finished run
number of examples = 781265
weighted example sum = 7.813e+05
weighted label sum = 0
average loss = 0.04582
best constant = 0
total feature number = 343993166
 7:09PM 1-of-3-10: vw --cb 2 --cb_type ips --ngram 2 --skips 4 -b 24 -l 0.25 rcv1_train.cb_vw -c
Generating 2-grams for all namespaces.
Generating 4-skips for all namespaces.
Num weight bits = 24
learning rate = 0.25
initial_t = 0
power_t = 0.5
using cache_file = rcv1_train.cb_vw.cache
ignoring text input in favor of cache input
num sources = 1
average     since            example     example  current  current  current
loss        last             counter      weight    label  predict features
*estimate* *estimate*                                                     avglossreg last pred  last correc
t
0.666667   0.666667           3        3.0    known         2       316   0.333333   0.000000   0.000000
0.333333   0.000000           6        6.0    known         2       160   0.333333   0.000000   1.000000
0.363636   0.400000          11       11.0    known         2       202   0.363636   0.000000   0.000000
0.454545   0.545455          22       22.0    known         2       502   0.363636   0.000000   0.000000
0.363636   0.272727          44       44.0    known         2       370   0.477273   0.000000   0.000000
0.275862   0.186047          87       87.0    known         1       340   0.471264   0.000000   1.000000
0.298851   0.321839         174      174.0    known         2       130   0.459770   0.000000   0.000000
0.212644   0.126437         348      348.0    known         2       262   0.465517   0.000000   1.000000
0.186782   0.160920         696      696.0    known         2       124   0.472701   0.000000   0.000000
0.150862   0.114943        1392     1392.0    known         1      1066   0.481322   0.000000   0.000000
0.128592   0.106322        2784     2784.0    known         1       280   0.497845   0.000000   1.000000
0.108836   0.089080        5568     5568.0    known         1       514   0.497306   0.000000   0.000000
0.094836   0.080833       11135    11135.0    known         2       352   0.502829   0.000000   1.000000
```

Terminal 1:27 7:09 PM John Langford

```
weighted label sum = 0
average loss = 0.04582
best constant = 0
total feature number = 343993166
 7:09PM 1-of-3-10: vw --cb 2 --cb_type ips --ngram 2 --skips 4 -b 24 -l 0.25 rcv1_train.cb_vw -c
Generating 2-grams for all namespaces.
Generating 4-skips for all namespaces.
Num weight bits = 24
learning rate = 0.25
initial_t = 0
power_t = 0.5
using cache_file = rcv1_train.cb_vw.cache
ignoring text input in favor of cache input
num sources = 1
average    since          example     example  current  current  current
loss       last           counter      weight    label  predict features
*estimate* *estimate*                                                    avglossreg last pred  last correc
t
0.666667   0.666667             3         3.0    known        2      316  0.333333   0.000000   0.000000
0.333333   0.000000             6         6.0    known        2      160  0.333333   0.000000   1.000000
0.363636   0.400000            11        11.0    known        2      202  0.363636   0.000000   0.000000
0.454545   0.545455            22        22.0    known        2      502  0.363636   0.000000   0.000000
0.363636   0.272727            44        44.0    known        2      370  0.477273   0.000000   0.000000
0.275862   0.186047            87        87.0    known        1      340  0.471264   0.000000   1.000000
0.298851   0.321839           174       174.0    known        2      130  0.459770   0.000000   0.000000
0.212644   0.126437           348       348.0    known        2      262  0.465517   0.000000   1.000000
0.186782   0.160920           696       696.0    known        2      124  0.472701   0.000000   0.000000
0.150862   0.114943          1392      1392.0    known        1     1066  0.481322   0.000000   0.000000
0.128592   0.106322          2784      2784.0    known        1      280  0.497845   0.000000   1.000000
0.108836   0.089080          5568      5568.0    known        1      514  0.497306   0.000000   0.000000
0.094836   0.080833         11135     11135.0    known        2      352  0.502829   0.000000   1.000000
0.085231   0.075624         22269     22269.0    known        2      820  0.504109   0.000000   0.000000
0.075488   0.065745         44537     44537.0    known        2      226  0.502997   0.000000   0.000000
0.069067   0.062646         89073     89073.0    known        1      142  0.502565   0.000000   1.000000
0.063094   0.057122        178146    178146.0    known        2      274  0.502107   0.000000   1.000000
0.058233   0.053372        356291    356291.0    known        1      580  0.500931   0.000000   1.000000
```

How do you train?

1. Learn $\hat{r}(a, x)$.
2. Compute for each x the double-robust estimate for each $a' \in \{1, \ldots, K\}$:

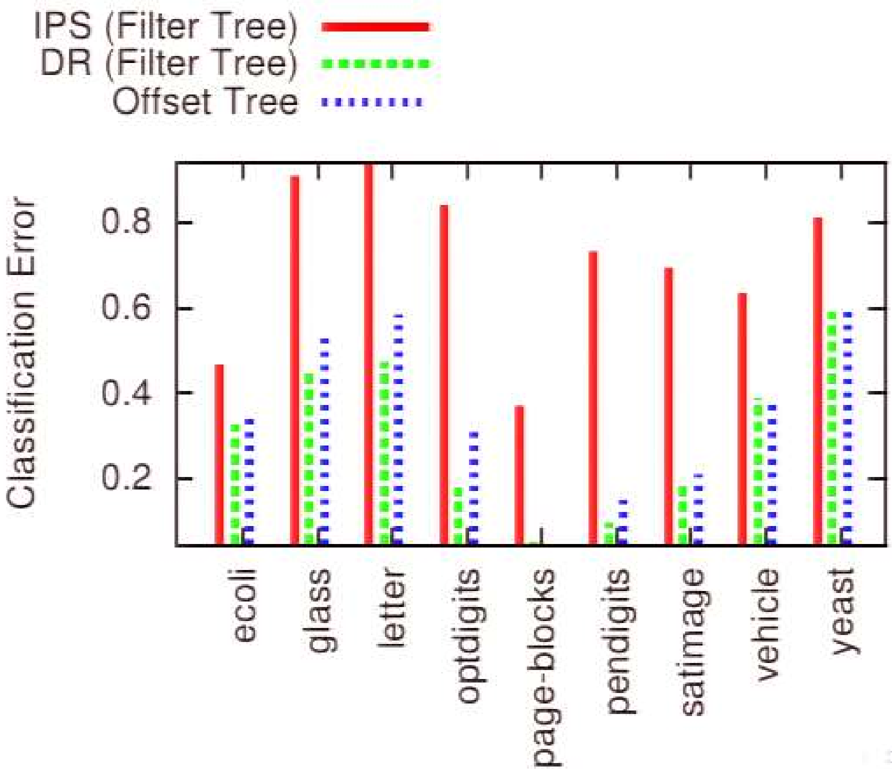
$$\frac{(r - \hat{r}(a, x))I(a' = a)}{p(a|x)} + \hat{r}(a', x)$$

3. Learn π using a cost-sensitive classifier. We'll use Vowpal Wabbit: http://hunch.net/~vw

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

Experimental Results

IPS = Inverse probability
DR = Doubly Robust, with $\hat{r}(a, x) = w_a \cdot x$
Filter Tree = Cost Sensitive Multiclass classifier
Offset Tree = Earlier method for CB learning with same representation



Summary of methods

1. Deployment. Aka A/B testing. Gold standard for measurement and cost.
2. Direct Method. Often used by people who don't know what they are doing. Some value when used in conjunction with careful exploration.
3. Inverse probability. Unbiased, but possibly high variance.
4. Inverse propensity score. For when you don't know or don't trust recorded probabilities. Debugging tool that gives hints, but caution is in order.
5. Offset Tree. (not discussed) Only known logarithmic time method.
6. Double robust. Best known offline method. Unbiased + reduced variance.

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 $\Pi = \{\pi : X \to A\}$:

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

What is exploration?

Exploration = Choosing not-obviously best actions to gather information for better performance in the future.

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 $\Pi = \{\pi : X \to A\}$:

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

What is exploration?

Exploration = Choosing not-obviously best actions to gather information for better performance in the future.

What is exploration?

Exploration = Choosing not-obviously best actions to gather information for better performance in the future.

There are two kinds:

1. Deterministic. Choose action A, then B, then C, then A, then B, ...
2. Randomized. Choose random actions according to some distribution over actions.

What is exploration?

Exploration = Choosing not-obviously best actions to gather information for better performance in the future.
There are two kinds:

1. Deterministic. Choose action A, then B, then C, then A, then B, ...
2. Randomized. Choose random actions according to some distribution over actions.

We discuss Randomized here.

1. There are no good deterministic exploration algorithms in this setting.
2. Supports off-policy evaluation. (See first half.)
3. Randomize = robust to delayed updates, which are very common in practice.

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 $\Pi = \{\pi : X \to A\}$:

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

What is exploration?

Exploration = Choosing not-obviously best actions to gather information for better performance in the future.

There are two kinds:

1. Deterministic. Choose action A, then B, then C, then A, then B, ...
2. Randomized. Choose random actions according to some distribution over actions.

What is exploration?

Exploration = Choosing not-obviously best actions to gather information for better performance in the future.

There are two kinds:

1. Deterministic. Choose action A, then B, then C, then A, then B, ...
2. Randomized. Choose random actions according to some distribution over actions.

We discuss Randomized here.

1. There are no good deterministic exploration algorithms in this setting.
2. Supports off-policy evaluation. (See first half.)
3. Randomize = robust to delayed updates, which are very common in practice.

Outline

1. Using Exploration
    1. Problem Definition
    2. Direct Method fails
    3. Importance Weighting
    4. Missing Probabilities
    5. Doubly Robust
2. Doing Exploration
    1. Exploration First
    2. ϵ-Greedy
    3. epoch Greedy
    4. Policy Elimination
    5. Thompson Sampling

Explore τ then Follow the Leader (Explore-τ)

Explore τ then Follow the Leader (Explore-τ)

Initially, $h = \emptyset$

For the first τ rounds

1. Observe x.
2. Choose a uniform randomly.
3. Observe r, and add (x, a, r) to h.

For the next T rounds, use empirical best.

Explore τ then Follow the Leader (Explore-τ)

Initially, $h = \emptyset$

For the first τ rounds

1. Observe x.
2. Choose a uniform randomly.
3. Observe r, and add (x, a, r) to h.

For the next T rounds, use empirical best.

Suppose all examples are drawn from a fixed distribution $D(x, \vec{r})$.

Theorem: For all D, Π, Explore-τ has regret $O\left(\frac{\tau}{T} + \sqrt{\frac{|A| \ln |\Pi|}{\tau}}\right)$ with high probability.

Explore τ then Follow the Leader (Explore-τ)

Initially, $h = \emptyset$

For the first τ rounds

1. Observe x.
2. Choose a uniform randomly.
3. Observe r, and add (x, a, r) to h.

For the next T rounds, use empirical best.

Suppose all examples are drawn from a fixed distribution $D(x, \vec{r})$.

Theorem: For all D, Π, Explore-τ has regret $O\left(\frac{\tau}{T} + \sqrt{\frac{|A| \ln |\Pi|}{\tau}}\right)$ with high probability.

Proof: After τ rounds, a large deviation bound implies empirical average value of a policy deviates from expectation $E_{(x,\vec{r})\sim D}[r_{\pi(x)}]$ by at most $\sqrt{\frac{|A| \ln(|\Pi|/\delta)}{\tau}}$, so regret is bounded by $\frac{\tau}{T} + \frac{T}{T}\sqrt{\frac{|A| \ln(|\Pi|/\delta)}{\tau}}$.

What does this mean?

1. +Easiest approach: offline prerecorded exploration can feed into any learning algorithm. See first half.
2. -Doesn't adapt when world changes.
3. -Underexploration common. Think of clinical trials.

Explore τ then Follow the Leader (Explore-τ)

Initially, $h = \emptyset$

For the first τ rounds

1. Observe x.
2. Choose a uniform randomly.
3. Observe r, and add (x, a, r) to h.

For the next T rounds, use empirical best.

Suppose all examples are drawn from a fixed distribution $D(x, \vec{r})$.

Theorem: For all D, Π, Explore-τ has regret $O\left(\frac{\tau}{T} + \sqrt{\frac{|A| \ln |\Pi|}{\tau}}\right)$ with high probability.

Proof: After τ rounds, a large deviation bound implies empirical average value of a policy deviates from expectation $E_{(x,\vec{r})\sim D}[r_{\pi(x)}]$ by at most $\sqrt{\frac{|A| \ln(|\Pi|/\delta)}{\tau}}$, so regret is bounded by
$\frac{\tau}{T} + \frac{T}{T}\sqrt{\frac{|A| \ln(|\Pi|/\delta)}{\tau}}$.

At optimal τ? $O\left(\left(\frac{|A| \ln |\Pi|}{T}\right)^{1/3}\right)$

What does this mean?

1. +Easiest approach: offline prerecorded exploration can feed into any learning algorithm. See first half.
2. -Doesn't adapt when world changes.
3. -Underexploration common. Think of clinical trials.

ϵ-Greedy

1. Observe x.
2. With probability $1 - \epsilon$
   1. Choose learned a
   2. Observe r, and learn with $(x, a, r, 1 - \epsilon)$.

ϵ-Greedy

1. Observe x.
2. With probability $1 - \epsilon$
   1. Choose learned a
   2. Observe r, and learn with $(x, a, r, 1 - \epsilon)$.

   With probability ϵ
   1. Choose Uniform random other a
   2. Observe r, and learn with $(x, a, r, \epsilon/(|A| - 1))$.

ϵ-Greedy

1. Observe x.
2. With probability $1 - \epsilon$
   1. Choose learned a
   2. Observe r, and learn with $(x, a, r, 1 - \epsilon)$.

   With probability ϵ
   1. Choose Uniform random other a
   2. Observe r, and learn with $(x, a, r, \epsilon/(|A| - 1))$.

Theorem: ϵ-Greedy has regret $O\left(\epsilon + \sqrt{\frac{|A| \ln |\Pi|}{T\epsilon}}\right)$

What does this mean?

1. -Harder Approach: Need online learning algorithm to use.
2. +Adapts when world changes.
3. -Overexploration common. Bad possibilities keep being explored.

What does this mean?

1. -Harder Approach: Need online learning algorithm to use.
2. +Adapts when world changes.
3. -Overexploration common. Bad possibilities keep being explored.

Can we do better?

Epoch Greedy

At every timestep t, the learned policy has an empirical performance known up to some precision ϵ_t which can be estimated.

Epoch Greedy

At every timestep t, the learned policy has an empirical performance known up to some precision ϵ_t which can be estimated.

1. Observe x.
2. With probability $1 - \epsilon_t$
   1. Choose learned a
   2. Observe r, update ϵ_t and learn with $(x, a, r, 1 - \epsilon_t)$.

Epoch Greedy

At every timestep t, the learned policy has an empirical performance known up to some precision ϵ_t which can be estimated.

1. Observe x.
2. With probability $1 - \epsilon_t$
   1. Choose learned a
   2. Observe r, update ϵ_t and learn with $(x, a, r, 1 - \epsilon_t)$.

   With probability ϵ_t
   1. Choose Uniform random other a
   2. Observe r, update ϵ_t and learn with $(x, a, r, \epsilon_t/(|A| - 1))$.

Epoch Greedy

At every timestep t, the learned policy has an empirical performance known up to some precision ϵ_t which can be estimated.

1. Observe x.
2. With probability $1 - \epsilon_t$
   1. Choose learned a
   2. Observe r, update ϵ_t and learn with $(x, a, r, 1 - \epsilon_t)$.

   With probability ϵ_t
   1. Choose Uniform random other a
   2. Observe r, update ϵ_t and learn with $(x, a, r, \epsilon_t/(|A| - 1))$.

Theorem: Epoch Greedy has regret $O\left(\left(\frac{|A| \ln |\Pi|}{T}\right)^{1/3}\right)$ with high probability.
Autotuning!

What does this mean?

1. -Harder Approach: Need online learning algorithm to use + keeping track of deviation bound.
2. +Adapts when world changes.
3. +Neither under nor over exploration.

What does this mean?

1. -Harder Approach: Need online learning algorithm to use + keeping track of deviation bound.
2. +Adapts when world changes.
3. +Neither under nor over exploration.

Is it possible to do better?

	Supervised	τ-first/ϵ-greedy/epoch-greedy
Regret	$O\left(\left(\frac{\ln \lvert\Pi\rvert}{T}\right)^{\frac{1}{2}}\right)$	$O\left(\left(\frac{\lvert A\rvert \ln \lvert\Pi\rvert}{T}\right)^{\frac{1}{3}}\right)$

Epoch Greedy

At every timestep t, the learned policy has an empirical performance known up to some precision ϵ_t which can be estimated.

1. Observe x.
2. With probability $1 - \epsilon_t$
    1. Choose learned a
    2. Observe r, update ϵ_t and learn with $(x, a, r, 1 - \epsilon_t)$.

   With probability ϵ_t
    1. Choose Uniform random other a
    2. Observe r, update ϵ_t and learn with $(x, a, r, \epsilon_t/(|A| - 1))$.

What does this mean?

1. -Harder Approach: Need online learning algorithm to use.
2. +Adapts when world changes.
3. -Overexploration common. Bad possibilities keep being explored.

Can we do better?

ϵ-Greedy

1. Observe x.
2. With probability $1 - \epsilon$
   1. Choose learned a
   2. Observe r, and learn with $(x, a, r, 1 - \epsilon)$.

   With probability ϵ
   1. Choose Uniform random other a
   2. Observe r, and learn with $(x, a, r, \epsilon/(|A| - 1))$.

Theorem: ϵ-Greedy has regret $O\left(\epsilon + \sqrt{\frac{|A| \ln |\Pi|}{T\epsilon}}\right)$

For optimal epsilon? $O\left(\left(\frac{|A| \ln |\Pi|}{T}\right)^{1/3}\right)$

What does this mean?

1. +Easiest approach: offline prerecorded exploration can feed into any learning algorithm. See first half.
2. -Doesn't adapt when world changes.
3. -Underexploration common. Think of clinical trials.

Can we do better?

Explore τ then Follow the Leader (Explore-τ)

Initially, $h = \emptyset$

For the first τ rounds

1. Observe x.
2. Choose a uniform randomly.
3. Observe r, and add (x, a, r) to h.

For the next T rounds, use empirical best.

Suppose all examples are drawn from a fixed distribution $D(x, \vec{r})$.

Theorem: For all $D, \Pi,$ Explore-τ has regret $O\left(\frac{\tau}{T} + \sqrt{\frac{|A| \ln |\Pi|}{\tau}}\right)$ with high probability.

Proof: After τ rounds, a large deviation bound implies empirical average value of a policy deviates from expectation $E_{(x,\vec{r})\sim D}[r_{\pi(x)}]$ by at most $\sqrt{\frac{|A| \ln(|\Pi|/\delta)}{\tau}}$, so regret is bounded by

$\frac{\tau}{T} + \frac{T}{T}\sqrt{\frac{|A| \ln(|\Pi|/\delta)}{\tau}}$.

At optimal τ? $O\left(\left(\frac{|A| \ln |\Pi|}{T}\right)^{1/3}\right)$

ϵ-Greedy

1. Observe x.
2. With probability $1 - \epsilon$
    1. Choose learned a
    2. Observe r, and learn with $(x, a, r, 1 - \epsilon)$.

   With probability ϵ
    1. Choose Uniform random other a
    2. Observe r, and learn with $(x, a, r, \epsilon/(|A| - 1))$.

Theorem: ϵ-Greedy has regret $O\left(\epsilon + \sqrt{\frac{|A| \ln |\Pi|}{T\epsilon}}\right)$

Epoch Greedy

At every timestep t, the learned policy has an empirical performance known up to some precision ϵ_t which can be estimated.

1. Observe x.
2. With probability $1 - \epsilon_t$
    1. Choose learned a
    2. Observe r, update ϵ_t and learn with $(x, a, r, 1 - \epsilon_t)$.

Epoch Greedy

At every timestep t, the learned policy has an empirical performance known up to some precision ϵ_t which can be estimated.

1. Observe x.
2. With probability $1 - \epsilon_t$
   1. Choose learned a
   2. Observe r, update ϵ_t and learn with $(x, a, r, 1 - \epsilon_t)$.

   With probability ϵ_t
   1. Choose Uniform random other a
   2. Observe r, update ϵ_t and learn with $(x, a, r, \epsilon_t/(|A| - 1))$.

Theorem: Epoch Greedy has regret $O\left(\left(\frac{|A| \ln |\Pi|}{T}\right)^{1/3}\right)$ with high probability.
Autotuning!

What does this mean?

1. -Harder Approach: Need online learning algorithm to use + keeping track of deviation bound.
2. +Adapts when world changes.
3. +Neither under nor over exploration.

What does this mean?

1. -Harder Approach: Need online learning algorithm to use + keeping track of deviation bound.
2. +Adapts when world changes.
3. +Neither under nor over exploration.

Is it possible to do better?

	Supervised	τ-first/ϵ-greedy/epoch-greedy
Regret	$O\left(\left(\frac{\ln \lvert\Pi\rvert}{T}\right)^{\frac{1}{2}}\right)$	$O\left(\left(\frac{\lvert A\rvert \ln \lvert\Pi\rvert}{T}\right)^{\frac{1}{3}}\right)$

Better 1: Policy Elimination

Policy_Elimination

Let $\Pi_0 = \Pi$ and $\mu_t = 1/\sqrt{Kt}$ and $\eta_t(\pi)$ =empirical reward
For each $t = 1, 2, \ldots$

1. Choose distribution P over Π_{t-1} s.t. for every remaining policy π, the expected variance of a value estimate is small.
2. observe x
3. Let $p(a)$ = fraction of P choosing a given x.
4. Choose $a \sim p$ and observe reward r
5. Let Π_t = remaining near empirical best policies.

Theorem: With high probability Policy_Elimination has regret

$$O\left(\sqrt{\frac{|A| \ln |\Pi|}{T}}\right)$$

Better 1: Policy Elimination

Policy_Elimination

Let $\Pi_0 = \Pi$ and $\mu_t = 1/\sqrt{Kt}$ and $\eta_t(\pi) =$empirical reward
For each $t = 1, 2, \ldots$

1. Choose distribution P over Π_{t-1} s.t. $\forall\ \pi \in \Pi_{t-1}$:
   $\mathbf{E}_{x \sim D_X}\left[\frac{1}{(1-K\mu_t)\Pr_{\pi' \sim P}(\pi'(x)=\pi(x))+\mu_t}\right] \leq 2K$
2. observe x
3. Let $p(a) =$ fraction of P choosing a given x.
4. Choose $a \sim p$ and observe reward r
5. Let $\Pi_t =$ remaining near empirical best policies.

Theorem: With high probability Policy_Elimination has regret

$$O\left(\sqrt{\frac{|A| \ln |\Pi|}{T}}\right)$$

Better 1: Policy Elimination

Policy_Elimination

Let $\Pi_0 = \Pi$ and $\mu_t = 1/\sqrt{Kt}$ and $\eta_t(\pi)$ =empirical reward
For each $t = 1, 2, \ldots$

1. Choose distribution P over Π_{t-1} s.t. $\forall\ \pi \in \Pi_{t-1}$:
   $\mathbf{E}_{x \sim D_X}\left[\frac{1}{(1-K\mu_t)\Pr_{\pi' \sim P}(\pi'(x)=\pi(x))+\mu_t}\right] \leq 2K$
2. observe x
3. Let $p(a) = (1 - K\mu_t)\Pr_{\pi \sim P}(\pi(x) = a) + \mu_t$
4. Choose $a \sim p$ and observe reward r
5. Let $\Pi_t = \{\pi \in \Pi_{t-1} : \eta_t(\pi) \geq \max_{\pi' \in \Pi_{t-1}} \eta_t(\pi') - K\mu_t\}$

Theorem: With high probability Policy_Elimination has regret

$$O\left(\sqrt{\frac{|A| \ln |\Pi|}{T}}\right)$$

What does this mean?

1. -Doesn't adapt when world changes.
2. ++Much more efficient exploration. Only efficient in special cases.
3. - -Much Harder Approach: Need to keep track of policies, which is often intractable.

Better 1: Policy Elimination

Policy_Elimination

Let $\Pi_0 = \Pi$ and $\mu_t = 1/\sqrt{Kt}$ and $\eta_t(\pi)$ =empirical reward
For each $t = 1, 2, \ldots$

1. Choose distribution P over Π_{t-1} s.t. $\forall\ \pi \in \Pi_{t-1}$:
   $\mathbf{E}_{x \sim D_X} \left[\frac{1}{(1-K\mu_t) \Pr_{\pi' \sim P}(\pi'(x)=\pi(x))+\mu_t} \right] \leq 2K$
2. observe x
3. Let $p(a) = (1 - K\mu_t) \Pr_{\pi \sim P}(\pi(x) = a) + \mu_t$
4. Choose $a \sim p$ and observe reward r
5. Let $\Pi_t = \{\pi \in \Pi_{t-1} : \eta_t(\pi) \geq \max_{\pi' \in \Pi_{t-1}} \eta_t(\pi') - K\mu_t\}$

Theorem: With high probability Policy_Elimination has regret

$$O\left(\sqrt{\frac{|A| \ln |\Pi|}{T}}\right)$$

What does this mean?

1. -Doesn't adapt when world changes.
2. ++Much more efficient exploration. Only efficient in special cases.
3. - -Much Harder Approach: Need to keep track of policies, which is often intractable.

What does this mean?

1. -Doesn't adapt when world changes.
2. ++Much more efficient exploration. Only efficient in special cases.
3. - -Much Harder Approach: Need to keep track of policies, which is often intractable.

Adapting algorithms exist (EXP4).
More efficient versions exist (RUCB), but not yet efficient enough.

Can you do better?

What does this mean?

1. -Doesn't adapt when world changes.
2. ++Much more efficient exploration. Only efficient in special cases.
3. - -Much Harder Approach: Need to keep track of policies, which is often intractable.

Adapting algorithms exist (EXP4).
More efficient versions exist (RUCB), but not yet efficient enough.

Better 1: Policy Elimination

Policy_Elimination

Let $\Pi_0 = \Pi$ and $\mu_t = 1/\sqrt{Kt}$ and $\eta_t(\pi)$ =empirical reward
For each $t = 1, 2, \ldots$

1. Choose distribution P over Π_{t-1} s.t. $\forall\ \pi \in \Pi_{t-1}$:
$\mathbf{E}_{x \sim D_X}\left[\frac{1}{(1-K\mu_t)\Pr_{\pi' \sim P}(\pi'(x)=\pi(x))+\mu_t}\right] \leq 2K$
2. observe x
3. Let $p(a) = (1 - K\mu_t)\Pr_{\pi \sim P}(\pi(x) = a) + \mu_t$
4. Choose $a \sim p$ and observe reward r
5. Let $\Pi_t = \{\pi \in \Pi_{t-1} : \eta_t(\pi) \geq \max_{\pi' \in \Pi_{t-1}} \eta_t(\pi') - K\mu_t\}$

Theorem: With high probability Policy_Elimination has regret

$$O\left(\sqrt{\frac{|A| \ln |\Pi|}{T}}\right)$$

What does this mean?

1. -Doesn't adapt when world changes.
2. ++Much more efficient exploration. Only efficient in special cases.
3. - -Much Harder Approach: Need to keep track of policies, which is often intractable.

Adapting algorithms exist (EXP4).
More efficient versions exist (RUCB), but not yet efficient enough.

Better 1: Policy Elimination

Policy_Elimination

Let $\Pi_0 = \Pi$ and $\mu_t = 1/\sqrt{Kt}$ and $\eta_t(\pi)$ =empirical reward
For each $t = 1, 2, \ldots$

1. Choose distribution P over Π_{t-1} s.t. $\forall\ \pi \in \Pi_{t-1}$:
   $\mathbf{E}_{x \sim D_X}\left[\frac{1}{(1-K\mu_t)\Pr_{\pi' \sim P}(\pi'(x)=\pi(x))+\mu_t}\right] \leq 2K$
2. observe x
3. Let $p(a) = (1 - K\mu_t)\Pr_{\pi \sim P}(\pi(x) = a) + \mu_t$
4. Choose $a \sim p$ and observe reward r
5. Let $\Pi_t = \{\pi \in \Pi_{t-1} : \eta_t(\pi) \geq \max_{\pi' \in \Pi_{t-1}} \eta_t(\pi') - K\mu_t\}$

Theorem: With high probability Policy_Elimination has regret

$$O\left(\sqrt{\frac{|A| \ln |\Pi|}{T}}\right)$$

What does this mean?

1. -Doesn't adapt when world changes.
2. ++Much more efficient exploration. Only efficient in special cases.
3. - -Much Harder Approach: Need to keep track of policies, which is often intractable.

Adapting algorithms exist (EXP4).
More efficient versions exist (RUCB), but not yet efficient enough.

Can you do better?

Can you do better?

Not in general.

Theorem: For all algorithms, there exists problems imposing regret:

$$\tilde{\Omega}\left(\sqrt{\frac{|A| \ln |\Pi|}{T}}\right)$$

Better 2: Thompson Sampling

Always maintain a Bayesian posterior over policies.
On each round sample policy from posterior, and act according to it.

Can you do better?

Better 2: Thompson Sampling

Always maintain a Bayesian posterior over policies.
On each round sample policy from posterior, and act according to it.

Better 2: Thompson Sampling

Always maintain a Bayesian posterior over policies.
On each round sample policy from posterior, and act according to it.
An efficient special case: Gaussian Posterior.

Thompson Sampling

Let $w =$ mean 0 multivariate gaussian.
For each $t = 1, 2, \ldots$

1. Draw $w' \sim w$
2. Observe x
3. Choose $a = \max_{a'} w' x_{a'}$
4. Observe reward r.
5. Bayesian update w with (x, a, r).

What does it mean?

1. +Efficient special cases for Gaussian posteriors.
2. +Known to work well empirically sometimes.
3. -Not robust to model misspecification: $\tilde{\Omega}\left(\frac{|\Pi|}{T}\right)$ regret.

The current state

Starter	
Baseline	
Purring	
Shiny	
Something to try	

The current state

Explore-τ	Simplest Possible
Baseline	
Purring	
Shiny	
Something to try	

The current state

Explore-τ	Simplest Possible
ϵ-Greedy	Simplest Adaptive
Purring	
Shiny	
Something to try	

The current state

Explore-τ	Simplest Possible
ϵ-Greedy	Simplest Adaptive
Epoch Greedy	Unequivocal Improvement
Shiny	
Something to try	

The current state

Explore-τ	Simplest Possible
ϵ-Greedy	Simplest Adaptive
Epoch Greedy	Unequivocal Improvement
Policy Elimination	Optimal Impractical
Something to try	

The current state

Explore-τ	Simplest Possible
ϵ-Greedy	Simplest Adaptive
Epoch Greedy	Unequivocal Improvement
Policy Elimination	Optimal Impractical
Thompson Sampling	Sometimes Excellent

The current state

Explore-τ	Simplest Possible
ϵ-Greedy	Simplest Adaptive
Epoch Greedy	Unequivocal Improvement
Policy Elimination	Optimal Impractical
Thompson Sampling	Sometimes Excellent

You can see the edge of the understood world here. We hope to see further soon.
Further discussion: http://hunch.net

Bibliography: Using Exploration

Inverse An old technique, not sure where it was first used.

Nonrand J. Langford, A. Strehl, and J. Wortman Exploration Scavenging ICML 2008.

Offset A. Beygelzimer and J. Langford, The Offset Tree for Learning with Partial Labels KDD 2009.

Implicit A. Strehl, J. Langford, S. Kakade, and L. Li Learning from Logged Implicit Exploration Data NIPS 2010.

DRobust M. Dudik, J. Langford and L. Li, Doubly Robust Policy Evaluation and Learning, ICML 2011.

Bibliography: Doing Exploration

Tau-first Unclear first use?

ϵ-Greedy Unclear first use?

Epoch J. Langford and T. Zhang, The Epoch-Greedy Algorithm for Contextual Multi-armed Bandits, NIPS 2007.

EXP4 P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. SIAM Journal of Computing, 32(1):48–77, 2002b.

EXP4++ B. McMahan and M. Streeter, Tighter bounds for Multi-Armed Bandits with Expert Advice, COLT 2009.

Bibliograph: Doing Exploration II

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

Realizable A. Agarwal, M. Dudik, S. Kale, and J. Langford, Contextual Bandit Learning with Predictable Rewards, AIStat 2012.

Thompson W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika, 25(3-4):285–294, 1933.

Prior fail Alina Beygelzimer, John Langford, Lihong Li, Lev Reyzin, Robert E. Schapire, An Optimal High Probability Algorithm for the Contextual Bandit Problem AIStat 2011.

Empirical O. Chapelle and L. Li. An Empirical Evaluation of Thompson Sampling, NIPS 2011.

The current state

Explore-τ	Simplest Possible
ϵ-Greedy	Simplest Adaptive
Epoch Greedy	Unequivocal Improvement
Policy Elimination	Optimal Impractical
Thompson Sampling	Sometimes Excellent

You can see the edge of the understood world here. We hope to see further soon.
Further discussion: http://hunch.net

Better 1: Policy Elimination

Policy_Elimination

Let $\Pi_0 = \Pi$ and $\mu_t = 1/\sqrt{Kt}$ and $\eta_t(\pi) =$empirical reward
For each $t = 1, 2, \ldots$

1. Choose distribution P over Π_{t-1} s.t. for every remaining policy π, the expected variance of a value estimate is small.
2. observe x
3. Let $p(a) =$ fraction of P choosing a given x.
4. Choose $a \sim p$ and observe reward r
5. Let $\Pi_t =$ remaining near empirical best policies.

Theorem: With high probability Policy_Elimination has regret

$$O\left(\sqrt{\frac{|A| \ln |\Pi|}{T}}\right)$$

ϵ-Greedy

1. Observe x.
2. With probability $1-\epsilon$
   1. Choose learned a
   2. Observe r, and learn with $(x, a, r, 1-\epsilon)$.

   With probability ϵ
   1. Choose Uniform random other a
   2. Observe r, and learn with $(x, a, r, \epsilon/(|A|-1))$.

Theorem: ϵ-Greedy has regret $O\left(\epsilon + \sqrt{\frac{|A| \ln |\Pi|}{T\epsilon}}\right)$

Explore τ then Follow the Leader (Explore-τ)

Initially, $h = \emptyset$

For the first τ rounds

1. Observe x.
2. Choose a uniform randomly.
3. Observe r, and add (x, a, r) to h.

For the next T rounds, use empirical best.

Suppose all examples are drawn from a fixed distribution $D(x, \vec{r})$.

Theorem: For all $D, \Pi,$ Explore-τ has regret $O\left(\frac{\tau}{T} + \sqrt{\frac{|A| \ln |\Pi|}{\tau}}\right)$ with high probability.

Proof: After τ rounds, a large deviation bound implies empirical average value of a policy deviates from expectation $E_{(x,\vec{r})\sim D}[r_{\pi(x)}]$ by at most $\sqrt{\frac{|A| \ln(|\Pi|/\delta)}{\tau}}$, so regret is bounded by $\frac{\tau}{T} + \frac{T}{T}\sqrt{\frac{|A| \ln(|\Pi|/\delta)}{\tau}}$.

What is exploration?

Exploration = Choosing not-obviously best actions to gather information for better performance in the future.

There are two kinds:

1. Deterministic. Choose action A, then B, then C, then A, then B, ...
2. Randomized. Choose random actions according to some distribution over actions.

We discuss Randomized here.

1. There are no good deterministic exploration algorithms in this setting.
2. Supports off-policy evaluation. (See first half.)
3. Randomize = robust to delayed updates, which are very common in practice.

What is exploration?

Exploration = Choosing not-obviously best actions to gather information for better performance in the future.

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 $\Pi = \{\pi : X \rightarrow A\}$:

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

Explore τ then Follow the Leader (Explore-τ)

Initially, $h = \emptyset$

For the first τ rounds

1. Observe x.
2. Choose a uniform randomly.
3. Observe r, and add (x, a, r) to h.

For the next T rounds, use empirical best.

Suppose all examples are drawn from a fixed distribution $D(x, \vec{r})$.

Theorem: For all D, Π, Explore-τ has regret $O\left(\frac{\tau}{T} + \sqrt{\frac{|A| \ln |\Pi|}{\tau}}\right)$ with high probability.

Proof: After τ rounds, a large deviation bound implies empirical average value of a policy deviates from expectation $E_{(x,\vec{r})\sim D}[r_{\pi(x)}]$ by at most $\sqrt{\frac{|A| \ln(|\Pi|/\delta)}{\tau}}$, so regret is bounded by $\frac{\tau}{T} + \frac{T}{T}\sqrt{\frac{|A| \ln(|\Pi|/\delta)}{\tau}}$.

At optimal τ?

ϵ-Greedy

1. Observe x.
2. With probability $1 - \epsilon$
    1. Choose learned a
    2. Observe r, and learn with $(x, a, r, 1 - \epsilon)$.

   With probability ϵ
    1. Choose Uniform random other a
    2. Observe r, and learn with $(x, a, r, \epsilon/(|A| - 1))$.

Epoch Greedy

At every timestep t, the learned policy has an empirical performance known up to some precision ϵ_t which can be estimated.

1. Observe x.
2. With probability $1 - \epsilon_t$
   1. Choose learned a
   2. Observe r, update ϵ_t and learn with $(x, a, r, 1 - \epsilon_t)$.

What does this mean?

1. -Harder Approach: Need online learning algorithm to use + keeping track of deviation bound.
2. +Adapts when world changes.
3. +Neither under nor over exploration.

Is it possible to do better?

	Supervised	τ-first/ϵ-greedy/epoch-greedy
Regret	$O\left(\left(\frac{\ln \lvert\Pi\rvert}{T}\right)^{\frac{1}{2}}\right)$	$O\left(\left(\frac{\lvert A\rvert \ln \lvert\Pi\rvert}{T}\right)^{\frac{1}{3}}\right)$

Better 1: Policy Elimination

Policy_Elimination

Let $\Pi_0 = \Pi$ and $\mu_t = 1/\sqrt{Kt}$ and $\eta_t(\pi) =$empirical reward
For each $t = 1, 2, \ldots$

1. Choose distribution P over Π_{t-1} s.t. for every remaining policy π, the expected variance of a value estimate is small.
2. observe x
3. Let $p(a) =$ fraction of P choosing a given x.
4. Choose $a \sim p$ and observe reward r
5. Let $\Pi_t =$ remaining near empirical best policies.

Theorem: With high probability Policy_Elimination has regret

$$O\left(\sqrt{\frac{|A| \ln |\Pi|}{T}}\right)$$

Better 1: Policy Elimination

Policy_Elimination

Let $\Pi_0 = \Pi$ and $\mu_t = 1/\sqrt{Kt}$ and $\eta_t(\pi)$ =empirical reward
For each $t = 1, 2, \ldots$

1. Choose distribution P over Π_{t-1} s.t. $\forall\, \pi \in \Pi_{t-1}$:
$\mathbf{E}_{x \sim D_X}\left[\frac{1}{(1-K\mu_t)\Pr_{\pi' \sim P}(\pi'(x)=\pi(x))+\mu_t}\right] \leq 2K$
2. observe x
3. Let $p(a) =$ fraction of P choosing a given x.
4. Choose $a \sim p$ and observe reward r
5. Let $\Pi_t =$ remaining near empirical best policies.

Theorem: With high probability Policy_Elimination has regret

$$O\left(\sqrt{\frac{|A| \ln |\Pi|}{T}}\right)$$

What does this mean?

1. -Doesn't adapt when world changes.
2. ++Much more efficient exploration. Only efficient in special cases.
3. - -Much Harder Approach: Need to keep track of policies, which is often intractable.

Better 1: Policy Elimination

Policy_Elimination

Let $\Pi_0 = \Pi$ and $\mu_t = 1/\sqrt{Kt}$ and $\eta_t(\pi)$ =empirical reward
For each $t = 1, 2, \ldots$

1. Choose distribution P over Π_{t-1} s.t. $\forall\ \pi \in \Pi_{t-1}$:
   $\mathbf{E}_{x\sim D_X}\left[\frac{1}{(1-K\mu_t)\Pr_{\pi'\sim P}(\pi'(x)=\pi(x))+\mu_t}\right] \leq 2K$
2. observe x
3. Let $p(a) = (1 - K\mu_t)\Pr_{\pi\sim P}(\pi(x) = a) + \mu_t$
4. Choose $a \sim p$ and observe reward r
5. Let $\Pi_t = \{\pi \in \Pi_{t-1} : \eta_t(\pi) \geq \max_{\pi'\in\Pi_{t-1}} \eta_t(\pi') - K\mu_t\}$

Theorem: With high probability Policy_Elimination has regret

$$O\left(\sqrt{\frac{|A| \ln |\Pi|}{T}}\right)$$

What does this mean?

1. -Harder Approach: Need online learning algorithm to use + keeping track of deviation bound.
2. +Adapts when world changes.
3. +Neither under nor over exploration.

Is it possible to do better?

	Supervised	τ-first/ϵ-greedy/epoch-greedy
Regret	$O\left(\left(\frac{\ln \lvert\Pi\rvert}{T}\right)^{\frac{1}{2}}\right)$	$O\left(\left(\frac{\lvert A\rvert \ln \lvert\Pi\rvert}{T}\right)^{\frac{1}{3}}\right)$

What is exploration?

Exploration = Choosing not-obviously best actions to gather information for better performance in the future.

There are two kinds:

1. Deterministic. Choose action A, then B, then C, then A, then B, ...
2. Randomized. Choose random actions according to some distribution over actions.

Summary of methods

1. Deployment. Aka A/B testing. Gold standard for measurement and cost.
2. Direct Method. Often used by people who don't know what they are doing. Some value when used in conjunction with careful exploration.
3. Inverse probability. Unbiased, but possibly high variance.
4. Inverse propensity score. For when you don't know or don't trust recorded probabilities. Debugging tool that gives hints, but caution is in order.
5. Offset Tree. (not discussed) Only known logarithmic time method.
6. Double robust. Best known offline method. Unbiased + reduced variance.

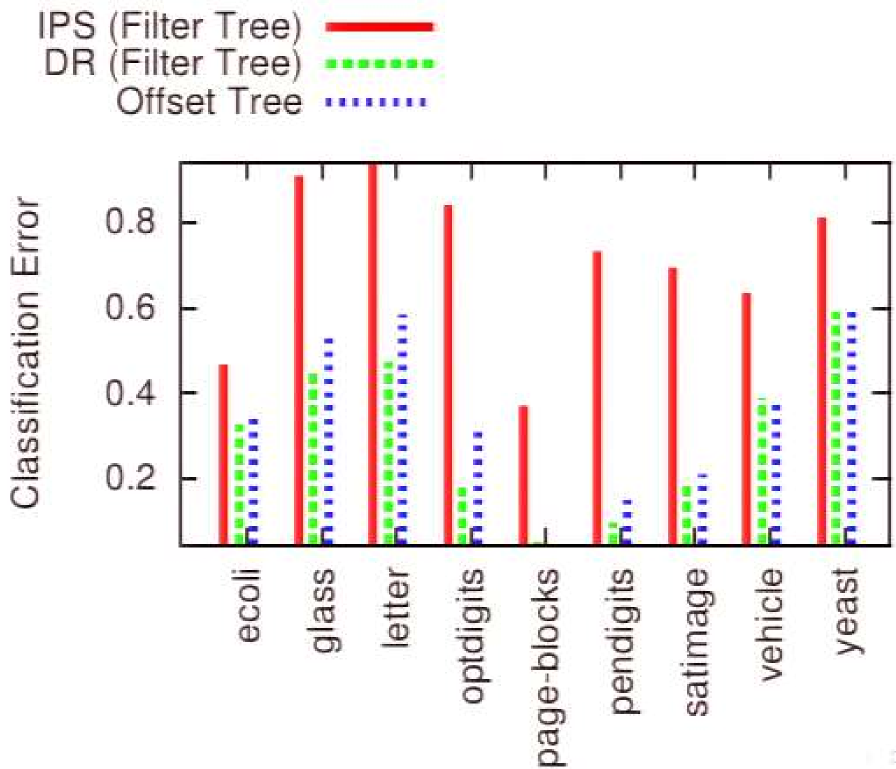
Experimental Results

IPS = Inverse probability

DR = Doubly Robust, with $\hat{r}(a, x) = w_a \cdot x$

Filter Tree = Cost Sensitive Multiclass classifier

Offset Tree = Earlier method for CB learning with same representation



How do you train?

1. Learn $\hat{r}(a, x)$.
2. Compute for each x the double-robust estimate for each $a' \in \{1, ..., K\}$:

$$\frac{(r - \hat{r}(a, x))I(a' = a)}{p(a|x)} + \hat{r}(a', x)$$

3. Learn π using a cost-sensitive classifier. We'll use Vowpal Wabbit: http://hunch.net/~vw

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

How do you test things?

Contextual Bandit datasets tend to be highly proprietary. What can you do?

1. Pick classification dataset.
2. Generate (x, a, r, p) quads via uniform random exploration of actions

Can we do better?

Suppose we have a (possibly bad) reward estimator $\hat{r}(a, x)$. How can we use it?

$$\text{Value'}(\pi) = \text{Average}\left(\frac{(r_a - \hat{r}(a,x))\mathbf{1}(\pi(x) = a)}{p_a} + \hat{r}(\pi(x), x)\right)$$

Let $\Delta(a, x) = \hat{r}(a, x) - E_{\vec{r}|x} r_a$ = reward deviation
Let $\delta(a, x) = 1 - \frac{p_a}{\hat{p}_a}$ = probability deviation

Theorem

For all policies π and all $(x, \vec{r})$:

$$|\text{Value'}(\pi) - E_{\vec{r}|x}[r_{\pi(x)}]| \leq |\Delta(\pi(x), x)\delta(\pi(x), x)|$$

The deviations multiply, so deviations < 1 means we win!

What if you don't know probabilities?

Suppose p was:

1. misrecorded "We randomized some actions, but then the Business Logic did something else."
2. not recorded "We randomized some scores which had an unclear impact on actions".
3. nonexistent "On Tuesday we did A and on Wednesday B".

Learn predictor $\hat{p}(a|x)$ on $(x, a)^*$ data.

Define new estimator: $\hat{V}(\pi) = \hat{E}_{x,a,r_a}\left[\frac{r_a I(\pi(x)=a)}{\max\{\tau, \hat{p}(a|x)\}}\right]$ where $\tau =$ small number.

Method 3: 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 of the form

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

where
$x =$ context
$a =$ action
$r_a =$ reward for action
$p_a =$ probability of action a
then evaluate:

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

Method 3: 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

Theorem

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

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

with deviations bounded by

$$O\left(\frac{1}{\sqrt{T \min_x p_{\pi(x)}}}\right)$$

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