Exploration in RL:
Contextual Bandit

Recap: MAB

Interactive learning process:

For $t = 0 \rightarrow T - 1$

(# based on historical information)

1. Learner pulls arm $I_t \in \{1,\ldots,K\}$

2. Learner observes an i.i.d reward $r_t \sim \nu_{I_t}$ of arm I_t

Recap: MAB

Interactive learning process:

For $t = 0 \to T - 1$

(# based on historical information)

1. Learner pulls arm $I_t \in \{1, \ldots, K\}$

2. Learner observes an i.i.d reward $r_t \sim \nu_{I_t}$ of arm I_t

Learning metric:

$$\text{Regret}_T = T\mu^\star - \sum_{t=0}^{T-1} \mu_{I_t}$$

Recap: MAB

Interactive learning process:

For $t = 0 \to T-1$

(# based on historical information)

1. Learner pulls arm $I_t \in \{1, \ldots, K\}$

2. Learner observes an i.i.d reward $r_t \sim \nu_{I_t}$ of arm I_t

Learning metric:

$$\text{Regret}_T = T\mu^\star - \sum_{t=0}^{T-1} \mu_{I_t}$$

Arm distributions are fixed across learning..

Question for Today:

Incorporate contexts into the interactive learning framework

Outline for today:

1. Introduction of the model

2. Algorithm

3. Theory and some practical considerations

Make the framework Context Dependent:

Interactive learning process:

For $t = 0 \rightarrow T - 1$

$\subseteq \mathbb{R}^d$

1. **A new context $x_t \in \mathscr{X}$ appears**

Make the framework Context Dependent:

Interactive learning process:

For $t = 0 \rightarrow T-1$

1. A new context $x_t \in \mathcal{X}$ appears

2. Learner picks action $a_t \in \mathcal{A}$

(# based on context x_t and historical information)

Make the framework Context Dependent:

Interactive learning process:

For $t = 0 \rightarrow T - 1$

1. **A new context $x_t \in \mathcal{X}$ appears**

   (# based on context x_t and historical information)

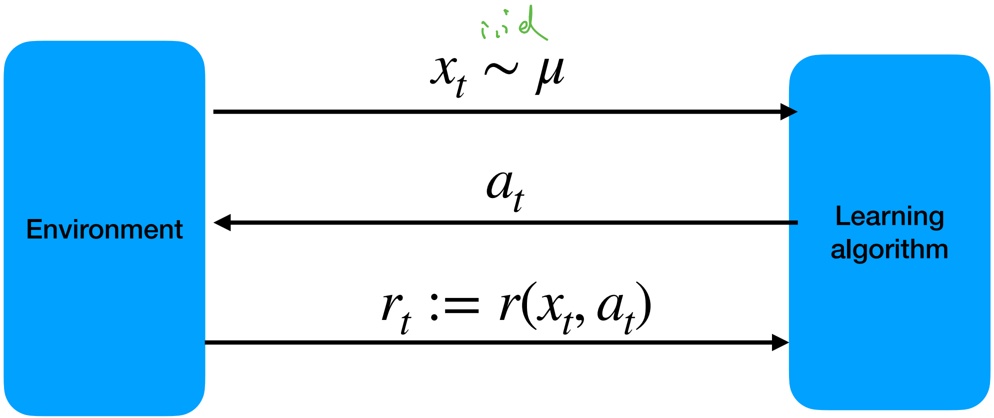
2. Learner picks action $a_t \in \mathcal{A}$

3. Learner observes an reward $r_t := r(x_t, a_t)$ Deterministic

Reward is context and arm dependent now!

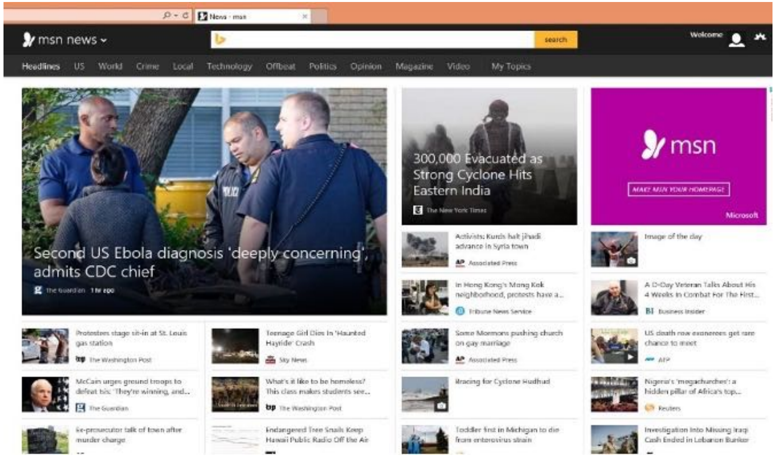
Make the framework Context Dependent:

Interactive learning process:



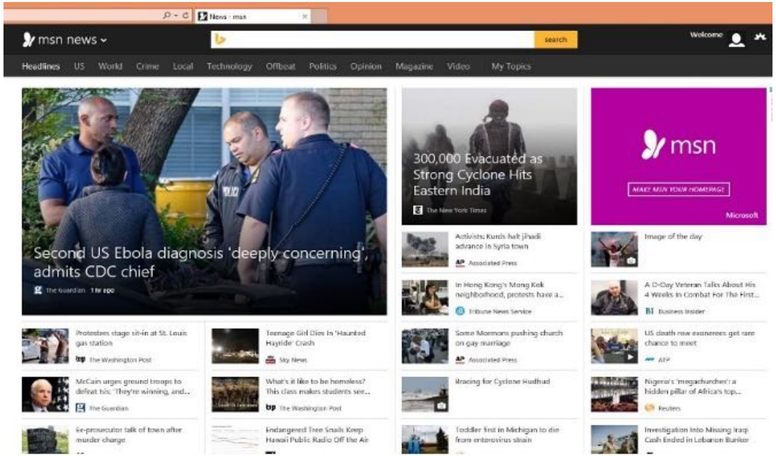
Examples:

Personalize recommendation system



Examples:

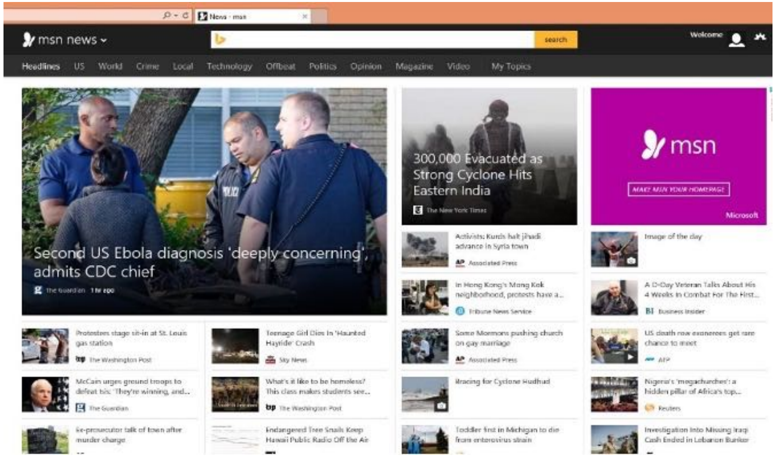
Personalize
recommendation system



Context: user's information (e.g., history ~~health conditions~~, age, height, weight, job type, etc)

Examples:

Personalize recommendation system

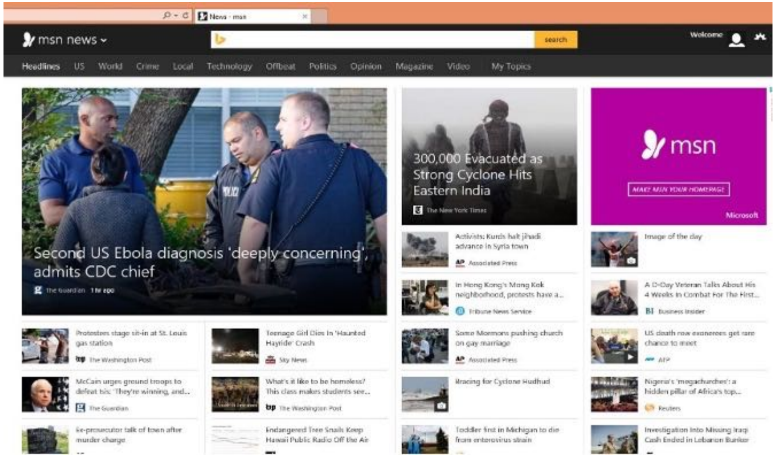


Context: user's information (e.g., history health conditions, age, height, weight, job type, etc)

Decisions (arms): news articles

Examples:

Personalize recommendation system



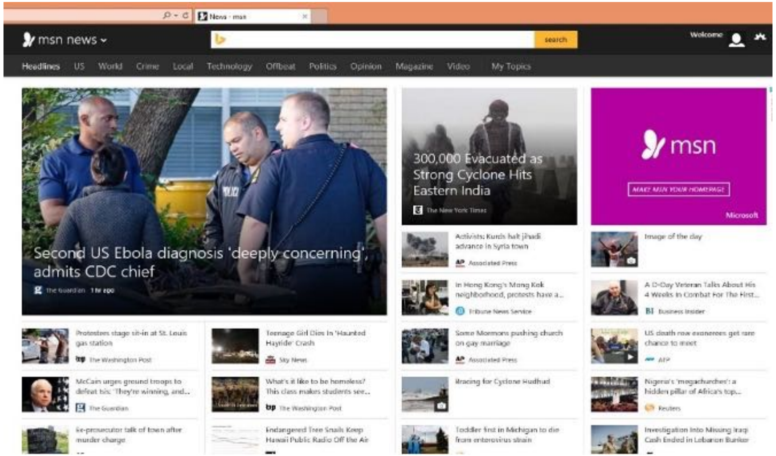
Context: user's information (e.g., history health conditions, age, height, weight, job type, etc)

Decisions (arms): news articles

Goal: learn to maximizes user click rate

Examples:

Personalize recommendation system



Context: user's information (e.g., history health conditions, age, height, weight, job type, etc)

Decisions (arms): news articles

Goal: learn to maximizes user click rate

Different users have different preferences on news, so need to personalize

Equivalently, it is an MDP with H = 1

Finite horizon MDP with $H = 1$

state

$$\mathscr{M} = \{\mathscr{X}, \mathscr{A}, r, H = 1, \mu\}$$

Equivalently, it is an MDP with H = 1

Finite horizon MDP with $H = 1$

$$\mathscr{M} = \{\mathscr{X}, \mathscr{A}, r, H = 1, \mu\}$$

Objective function:

$$\max_{\pi \in \Pi} \mathbb{E}_{x \sim \mu} \left[r(x, \pi(x)) \right]$$

Equivalently, it is an MDP with H = 1

Finite horizon MDP with $H = 1$

$$\mathcal{M} = \{\mathcal{X}, \mathcal{A}, r, H = 1, \mu\}$$

Objective function:

$$\max_{\pi \in \Pi} \mathbb{E}_{x \sim \mu} \left[r(x, \underbrace{\pi(x)}_{A}) \right]$$

For simplicity, we assume reward is deterministic;
The challenge is really from randomness in contexts

The Regret Metric

Fix a policy class Π (think about π as a classier from $x \to a$)

$\in \mathcal{X}$ $\in \mathcal{A}$

Denote optimal policy $\pi^\star = \arg\max_{\pi \in \Pi} \mathbb{E}_{x \sim \mu} r(x, \pi(x))$

The Regret Metric

Fix a policy class Π (think about π as a classier from $x \to a$)

Denote optimal policy $\pi^\star = \arg\max_{\pi \in \Pi} \mathbb{E}_{x \sim \mu} r(x, \pi(x))$

Every iteration, learner has a policy $\pi^t \in \Pi$
(Recommends $a_t = \pi^t(x_t)$, receives reward $r_t := r(x_t, \pi(x_t))$)

The Regret Metric

Fix a policy class Π (think about π as a classier from $x \to a$)

Denote optimal policy $\pi^\star = \arg\max_{\pi \in \Pi} \mathbb{E}_{x \sim \mu} r(x, \pi(x))$

Every iteration, learner has a policy $\pi^t \in \Pi$
(Recommends $a_t = \pi^t(x_t)$, receives reward $r_t := r(x_t, \pi(x_t))$)

$$\text{Regret}_T = T\mathbb{E}_{x \sim \mu}[r(x, \pi^\star(x))] - \sum_{t=0}^{T-1} \mathbb{E}_{x \sim \mu}[r(x, \pi^t(x))]$$

The Regret Metric

Fix a policy class Π (think about π as a classier from $x \to a$)

Denote optimal policy $\pi^\star = \arg\max_{\pi \in \Pi} \mathbb{E}_{x\sim\mu} r(x, \pi(x))$

Every iteration, learner has a policy $\pi^t \in \Pi$

(Recommends $a_t = \pi^t(x_t)$, receives reward $r_t := r(x_t, \pi(x_t))$)

$$\text{Regret}_T = T\mathbb{E}_{x\sim\mu}[r(x, \pi^\star(x))] - \sum_{t=0}^{T-1} \mathbb{E}_{x\sim\mu}[r(x, \pi^t(x))]$$

Total expected reward if we always uses $\pi^\star$ to recommend

The Regret Metric

Fix a policy class Π (think about π as a classier from $x \to a$)

Denote optimal policy $\pi^\star = \arg\max_{\pi \in \Pi} \mathbb{E}_{x\sim\mu} r(x, \pi(x))$

Every iteration, learner has a policy $\pi^t \in \Pi$

(Recommends $a_t = \pi^t(x_t)$, receives reward $r_t := r(x_t, \pi(x_t))$)

$$\text{Regret}_T = T\mathbb{E}_{x\sim\mu}[r(x, \pi^\star(x))] - \sum_{t=0}^{T-1} \mathbb{E}_{x\sim\mu}[r(x, \pi^t(x))]$$

Total expected reward if we always uses $\pi^\star$ to recommend

Total expected reward of our learned sequence of policies

Outline for today:

1. Introduction of the model

2. Algorithm

3. Theory and some practical considerations

Ingredient 1: Importance Weighting

The key challenging here is that we observe $r_t := r(x_t, a_t)$,
but we do not know $r(x_t, a)$ for $a \neq a_t$

Ingredient 1: Importance Weighting

The key challenging here is that we observe $r_t := r(x_t, a_t)$,
but we do not know $r(x_t, a)$ for $a \neq a_t$

Importance weighting actually allows us to get
unbiased estimate for ALL actions!

Ingredient 1: Importance Weighting

The key challenging here is that we observe $r_t := r(x_t, a_t)$,
but we do not know $r(x_t, a)$ for $a \neq a_t$

Importance weighting actually allows us to get
unbiased estimate for ALL actions!

Assume $a_t \sim p$ ($p \in \Delta(\mathscr{A})$), and we log $p(a_t)$, receive $r_t = r(x_t, a_t)$,

Ingredient 1: Importance Weighting

The key challenging here is that we observe $r_t := r(x_t, a_t)$,
but we do not know $r(x_t, a)$ for $a \neq a_t$

Importance weighting actually allows us to get
unbiased estimate for ALL actions!

Assume $a_t \sim p$ ($p \in \Delta(\mathscr{A})$), and we log $p(a_t)$, receive $r_t = r(x_t, a_t)$,

For all $a \in \mathscr{A}$, define $\hat{\mathbf{r}}[a] = \dfrac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)}$,

Ingredient 1: Importance Weighting

The key challenging here is that we observe $r_t := r(x_t, a_t)$,
but we do not know $r(x_t, a)$ for $a \neq a_t$

Importance weighting actually allows us to get
unbiased estimate for ALL actions!

Assume $a_t \sim p$ ($p \in \Delta(\mathscr{A})$), and we log $p(a_t)$, receive $r_t = r(x_t, a_t)$,

For all $a \in \mathscr{A}$, define $\hat{\mathbf{r}}[a] = \dfrac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)}$,

$$\hat{\mathbf{r}} := \begin{bmatrix} 0 \\ 0 \\ \cdots \\ r_t/p(a_t) \\ 0, \\ \cdots \\ 0 \end{bmatrix} \leftarrow a_t$$

Ingredient 1: Importance Weighting

The key challenging here is that we observe $r_t := r(x_t, a_t)$,
but we do not know $r(x_t, a)$ for $a \neq a_t$

Importance weighting actually allows us to get
unbiased estimate for ALL actions!

Assume $a_t \sim p$ ($p \in \Delta(\mathscr{A})$), and we log $p(a_t)$, receive $r_t = r(x_t, a_t)$,

For all $a \in \mathscr{A}$, define $\hat{\mathbf{r}}[a] = \dfrac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)}$,

$$\hat{\mathbf{r}} := \begin{bmatrix} 0 \\ 0 \\ \cdots \\ r_t / p(a_t) \\ 0, \\ \cdots \\ 0 \end{bmatrix}$$

$E[\hat{r}] = r(x_t, \cdot)$

$$\mathbb{E}_{a_t \sim p} \hat{\mathbf{r}}[a] = r(x_t, a), \forall a \in \mathscr{A}$$

Proving Importance Weighting

Assume $a_t \sim p$ ($p \in \Delta(\mathscr{A})$), and we log $p_t(a)$, receive $r_t = r(x_t, a_t)$,

For all $a \in \mathscr{A}$, define $\hat{\mathbf{r}}[a] = \dfrac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)}$, we have: $\mathbb{E}_{a_t \sim p}\hat{\mathbf{r}}[a] = r(x_t, a), \forall a \in \mathscr{A}$

Proving Importance Weighting

Assume $a_t \sim p$ ($p \in \Delta(\mathscr{A})$), and we log $p_t(a)$, receive $r_t = r(x_t, a_t)$,

For all $a \in \mathscr{A}$, define $\underline{\hat{\mathbf{r}}[a]} = \dfrac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)}$, we have: $\mathbb{E}_{a_t \sim p}\hat{\mathbf{r}}[a] = r(x_t, a), \forall a \in \mathscr{A}$

Randomness is from $\underline{a_t}$

Consider any $a \in \mathscr{A}$:

Proving Importance Weighting

Assume $a_t \sim p$ ($p \in \Delta(\mathscr{A})$), and we log $p_t(a)$, receive $r_t = r(x_t, a_t)$,

For all $a \in \mathscr{A}$, define $\hat{\mathbf{r}}[a] = \dfrac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)}$, we have: $\mathbb{E}_{a_t \sim p}\hat{\mathbf{r}}[a] = r(x_t, a), \forall a \in \mathscr{A}$

Consider any $a \in \mathscr{A}$:

$$\mathbb{E}_{a_t \sim p}\left[\hat{r}[a]\right] := r(x_t, a)$$

$$\mathbb{E}_{a_t \sim p}\frac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)} = \sum_{a_t \in \mathscr{A}} p(a_t)\frac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)}$$

$$= \mathbb{E}_{a_t \sim p}$$

Proving Importance Weighting

Assume $a_t \sim p$ ($p \in \Delta(\mathscr{A})$), and we log $p_t(a)$, receive $r_t = r(x_t, a_t)$,

For all $a \in \mathscr{A}$, define $\hat{\mathbf{r}}[a] = \dfrac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)}$, we have: $\mathbb{E}_{a_t \sim p}\hat{\mathbf{r}}[a] = r(x_t, a), \forall a \in \mathscr{A}$

Consider any $a \in \mathscr{A}$:

$$\mathbb{E}_{a_t \sim p}\frac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)} = \sum_{a_t \in \mathscr{A}} p(a_t)\frac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)}$$

← fires when $a_t = a$

$$= \cancel{p(a)}\frac{r(x_t, a)}{\cancel{p(a)}} = r(x_t, a)$$

$$\mathbb{E}_{a_t \sim p}\left[\hat{r}[a]\right] = r(x_t, a)$$

Ingredient 2: Reward-sensitive Classification Oracle

Recall classic classification:

Ingredient 2: Reward-sensitive Classification Oracle

Recall classic classification:

$$\{x_i, y_i\}_{i=1}^N, \quad x_i \in \mathcal{X}, y_i \in \{1, \ldots, |\mathcal{A}|\}$$

feature

label

Ingredient 2: Reward-sensitive Classification Oracle

Recall classic classification:

$$\{x_i, y_i\}_{i=1}^N, \quad x_i \in \mathcal{X}, y_i \in \{1, \ldots, |\mathcal{A}|\}$$

$$\arg\max_{\pi \in \Pi} \sum_{i=1}^N \mathbf{1}\{\pi(x_i) = y_i\}$$

$$\max_{\pi} \sum_{i=1}^N \mathbb{1}(\pi(x_i) = y_i)$$

$$\Leftrightarrow \min_{\pi} \sum_{i=1}^N \mathbb{1}(\pi(x_i) \neq y_i)$$

Ingredient 2: Reward-sensitive Classification Oracle

Recall classic classification:

$$\{x_i, y_i\}_{i=1}^N, \quad x_i \in \mathcal{X}, y_i \in \{1, \ldots, |\mathcal{A}|\}$$

$$\arg\max_{\pi \in \Pi} \sum_{i=1}^{N} \mathbf{1}\{\pi(x_i) = y_i\}$$

Let's generalize it to <u>R</u>eward-<u>S</u>ensitive <u>C</u>lassification:

Ingredient 2: Reward-sensitive Classification Oracle

Recall classic classification:

$$\{x_i, y_i\}_{i=1}^N, \quad x_i \in \mathcal{X}, y_i \in \{1, \ldots, |\mathcal{A}|\}$$

$$\arg\max_{\pi \in \Pi} \sum_{i=1}^N \mathbf{1}\{\pi(x_i) = y_i\}$$

Let's generalize it to <u>R</u>eward-<u>S</u>ensitive <u>C</u>lassification:

$$\{x_i, \mathbf{r}_i\}_{i=1}^N, \quad \underset{\substack{\uparrow \\ \text{feature}}}{x_i} \in \mathcal{X}, \mathbf{r}_i \in [0,1]^{|\mathcal{A}|}$$

Ingredient 2: Reward-sensitive Classification Oracle

Recall classic classification:

$$\{x_i, y_i\}_{i=1}^N, \quad x_i \in \mathcal{X}, y_i \in \{1, \ldots, |\mathcal{A}|\}$$

$$\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \leftarrow y_i$$

$$\arg\max_{\pi \in \Pi} \sum_{i=1}^N \mathbf{1}\{\pi(x_i) = y_i\}$$

Let's generalize it to <u>R</u>eward-<u>S</u>ensitive <u>C</u>lassification:

$$\{x_i, \mathbf{r}_i\}_{i=1}^N, \quad x_i \in \mathcal{X}, \mathbf{r}_i \in [0,1]^{|\mathcal{A}|}$$

Reward for label 1

$$\left(\begin{array}{c} x, \\ a \end{array} \quad \begin{bmatrix} r[1] \\ r[2] \\ \ldots \\ r[|\mathcal{A}|] \end{bmatrix} \right)$$

Ingredient 2: Reward-sensitive Classification Oracle

Recall classic classification:

$$\{x_i, y_i\}_{i=1}^N, \quad x_i \in \mathcal{X}, y_i \in \{1, \ldots, |\mathcal{A}|\}$$

$$\arg\max_{\pi \in \Pi} \sum_{i=1}^{N} \mathbf{1}\{\pi(x_i) = y_i\}$$

$r[:]$

Let's generalize it to Reward-Sensitive Classification:

$$\{x_i, \mathbf{r}_i\}_{i=1}^N, \quad x_i \in \mathcal{X}, \mathbf{r}_i \in [0,1]^{|\mathcal{A}|}$$

$$\left(x, \quad \begin{bmatrix} r[1] \\ r[2] \\ \cdots \\ r[|\mathcal{A}|] \end{bmatrix} \right)$$

$$\arg\max_{\pi \in \Pi} \sum_{i=1}^{N} \mathbf{r}_i[\pi(x_i)]$$

Ingredient 2: Reward-sensitive Classification Oracle

Recall classic classification:

$$\{x_i, y_i\}_{i=1}^N, \quad x_i \in \mathcal{X}, y_i \in \{1, \ldots, |\mathcal{A}|\}$$

$$\arg\max_{\pi \in \Pi} \sum_{i=1}^N \mathbf{1}\{\pi(x_i) = y_i\}$$

Let's generalize it to Reward-Sensitive Classification:

$$\{x_i, \mathbf{r}_i\}_{i=1}^N, \quad x_i \in \mathcal{X}, \mathbf{r}_i \in [0,1]^{|\mathcal{A}|}$$

$$\left(x, \quad \begin{bmatrix} r[1] \\ r[2] \\ \cdots \\ r[|\mathcal{A}|] \end{bmatrix} \right)$$

$$\arg\max_{\pi \in \Pi} \sum_{i=1}^N \mathbf{r}_i[\pi(x_i)]$$

We will do reduction to RSC

Summary so far:

1. Importance Weighting: we "magically" get unbiased estimate for all actions!

Assume $a_t \sim p$ ($p \in \Delta(\mathscr{A})$), For all $a \in \mathscr{A}$, define $\hat{\mathbf{r}}[a] = \dfrac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)}$, we have:

$$\hat{\mathbf{r}} := \begin{bmatrix} 0 \\ 0 \\ \cdots \\ r_t / p(a_t) \\ 0, \\ \cdots \\ 0 \end{bmatrix}$$

$$\mathbb{E}_{a_t \sim p} \hat{\mathbf{r}}[a] = r(x_t, a), \forall a \in \mathscr{A}$$

Summary so far:

1. Importance Weighting: we "magically" get unbiased estimate for all actions!

Assume $a_t \sim p$ ($p \in \Delta(\mathscr{A})$), For all $a \in \mathscr{A}$, define $\hat{\mathbf{r}}[a] = \dfrac{r(x_t, a)\mathbf{1}[a = a_t]}{p(a_t)}$, we have:

$$\hat{\mathbf{r}} := \begin{bmatrix} 0 \\ 0 \\ \dots \\ r_t / p(a_t) \\ 0, \\ \dots \\ 0 \end{bmatrix}$$

$$\mathbb{E}_{a_t \sim p} \hat{\mathbf{r}}[a] = r(x_t, a), \forall a \in \mathscr{A}$$

2. Reward-Sensitive Classification:

$$\{x_i, \mathbf{r}_i\}_{i=1}^N, \quad x_i \in \mathscr{X}, \mathbf{r}_i \in [0,1]^{|\mathscr{A}|}$$

$$\left(x, \quad \begin{bmatrix} r[1] \\ r[2] \\ \dots \\ r[|\mathscr{A}|] \end{bmatrix} \right)$$

$$\arg\max_{\pi \in \Pi} \sum_{i=1}^N \mathbf{r}_i[\pi(x_i)]$$

Put things together: Explore and Commit

For $t = 0 \to N - 1$:   (# exploration phase)

Put things together: Explore and Commit

For $t = 0 \rightarrow N - 1$:   (# exploration phase)

1. Observe $x_t \sim \mu$

Put things together: Explore and Commit

For $t = 0 \to N-1$: (# exploration phase)

uniform Dist over $\mathcal{A}$

1. Observe $x_t \sim \mu$

2. **Uniform-randomly** sample $a_t \sim \text{Unif}(\mathscr{A})$, receive reward $r_t = r(x_t, a_t)$

Put things together: Explore and Commit

For $t = 0 \rightarrow N-1$: (# exploration phase)

1. Observe $x_t \sim \mu$

2. **Uniform-randomly** sample $a_t \sim \text{Unif}(\mathscr{A})$, receive reward $r_t = r(x_t, a_t)$

All $a \in \mathcal{A}$

3. Use **IW**, form unbiased estimate $\hat{\mathbf{r}}_t[a] = \begin{cases} 0 & a \neq a_t \\ \dfrac{r_t}{1/|\mathscr{A}|} & a = a_t \end{cases}$

Put things together: Explore and Commit

For $t = 0 \rightarrow N-1$:   (# exploration phase)

1. Observe $x_t \sim \mu$

2. **Uniform-randomly** sample $a_t \sim \text{Unif}(\mathscr{A})$, receive reward $r_t = r(x_t, a_t)$

3. Use **IW**, form unbiased estimate $\hat{\mathbf{r}}_t[a] = \begin{cases} 0 & a \neq a_t \\ \dfrac{r_t}{1/|\mathscr{A}|} & a = a_t \end{cases}$

$\{x_i, \hat{r}_i\}_{i=0}^{N-1}$ ← Dataset for RSC

$\hat{r}_i \in \mathbb{R}^{|A|}$

$$\hat{\mathbf{r}} := \begin{bmatrix} 0 \\ 0 \\ \dots \\ r_t/p(a_t) \\ 0, \\ \dots \\ 0 \end{bmatrix}$$

$p(a_t) = 1/|A|$

Put things together: Explore and Commit

For $t = 0 \to N - 1:$ (# exploration phase)

1. Observe $x_t \sim \mu$

2. **Uniform-randomly** sample $a_t \sim \text{Unif}(\mathscr{A})$, receive reward $r_t = r(x_t, a_t)$

3. Use **IW**, form unbiased estimate $\hat{\mathbf{r}}_t[a] = \begin{cases} 0 & a \neq a_t \\ \frac{r_t}{1/|\mathscr{A}|} & a = a_t \end{cases}$

$$\hat{\mathbf{r}} := \begin{bmatrix} 0 \\ 0 \\ \cdots \\ r_t/p(a_t) \\ 0, \\ \cdots \\ 0 \end{bmatrix}$$

Call RSC oracle: $\hat{\pi} = \arg\max_{\pi \in \Pi} \boxed{\sum_{i=0}^{N-1} \hat{\mathbf{r}}_i[\pi(x_i)]}$

$x_t \sim \mu$

$\hat{r}_t$ unbiased estimate of $r(x_t, \cdot)$

for π:

$\frac{1}{N} \sum_{i=1}^{N} \hat{r}_i[\pi(x_i)]$

is unbiased est of $\mathbb{E}_{x \sim \mu}[r(x, \pi(x))]$

$\hookrightarrow \approx \mathbb{E}_{x \sim \mu}[r(x, \pi(x))]$

Put things together: Explore and Commit

For $t = 0 \rightarrow N - 1$: (# exploration phase)

1. Observe $x_t \sim \mu$

2. **Uniform-randomly** sample $a_t \sim \text{Unif}(\mathscr{A})$, receive reward $r_t = r(x_t, a_t)$

3. Use **IW**, form unbiased estimate $\hat{\mathbf{r}}_t[a] = \begin{cases} 0 & a \neq a_t \\ \frac{r_t}{1/|\mathscr{A}|} & a = a_t \end{cases}$

$$\hat{\mathbf{r}} := \begin{bmatrix} 0 \\ 0 \\ \dots \\ r_t/p(a_t) \\ 0, \\ \dots \\ 0 \end{bmatrix}$$

$E[\hat{r}_t] = r(x_t, \cdot)$

Call RSC oracle: $\hat{\pi} = \arg\max_{\pi \in \Pi} \sum_{i=0}^{N-1} \hat{\mathbf{r}}_t[\pi(x_t)] / N$

For $t = N \rightarrow T - 1$: (# exploitation phase)

Put things together: Explore and Commit

For $t = 0 \rightarrow N-1$: (# exploration phase)

1. Observe $x_t \sim \mu$

2. **Uniform-randomly** sample $a_t \sim \text{Unif}(\mathscr{A})$, receive reward $r_t = r(x_t, a_t)$

3. Use **IW**, form unbiased estimate $\hat{\mathbf{r}}_t[a] = \begin{cases} 0 & a \neq a_t \\ \frac{r_t}{1/|\mathscr{A}|} & a = a_t \end{cases}$

$$\hat{\mathbf{r}} := \begin{bmatrix} 0 \\ 0 \\ \dots \\ r_t/p(a_t) \\ 0, \\ \dots \\ 0 \end{bmatrix}$$

Call RSC oracle: $\hat{\pi} = \arg\max_{\pi \in \Pi} \sum_{i=0}^{N-1} \hat{\mathbf{r}}_t[\pi(x_t)]$

$\rightarrow \left[\frac{1}{N} \sum_{i=0}^{N-1} \hat{r}_t(\pi(x_t)) \right]$

$\Leftrightarrow \mathbb{E}_{x \sim \mu} \left[r(x, \pi(x)) \right]$

For $t = N \rightarrow T-1$: (# exploitation phase)

1. Observe $x_t \sim \mu$, and play $a_t = \hat{\pi}(x_t)$

Outline for today:

1. Introduction of the model

2. Algorithm

3. Theory and some practical considerations

Theory of the Explore and Commit Algorithm

For simplicity, assume Π is discrete (but could be exponential large)

a class of classifiers

Theory of the Explore and Commit Algorithm

For simplicity, assume Π is discrete (but could be exponential large)

[Theorem—informal] W/ high probability, properly setting the hyper-parameter N, Explore-and-Commit has the following regret:

$$\text{Regret}_T = T\mathbb{E}_{x\sim\mu}[r(x,\pi^\star(x))] - \sum_{t=0}^{T-1}\mathbb{E}_{x\sim\mu}[r(x,\pi^t(x))] = O\left(\underline{T^{2/3}K^{1/3}} \cdot \underline{\ln(|\Pi|)^{1/3}}\right)$$

$\text{VC-Dim}(\Pi)$

$$\frac{\text{Regret}_T}{T} \approx K^{\frac{1}{3}}\, T^{-\frac{1}{3}}$$

Practical Consideration

Instead of setting a hard threshold for explore and commit, we often interleave explore and exploitation

Practical Consideration

Instead of setting a hard threshold for explore and commit, we often interleave explore and exploitation

$\epsilon \in (0, 1)$

ϵ-greedy:

Every iteration t:

With probability $1 - \epsilon$, we play $a_t = \pi^t(x_t)$, ← Exploitation

and w/ probability ϵ, we play $a_t \sim \text{Unif}(\mathcal{A})$ ← Exploration

Practical Consideration

Instead of setting a hard threshold for explore and commit, we often interleave explore and exploitation

ϵ-greedy:

Every iteration t:

With probability $1 - \epsilon$, we play $a_t = \pi^t(x_t)$,

and w/ probability ϵ, we play $a_t \sim$ Unif$(\mathscr{A})$

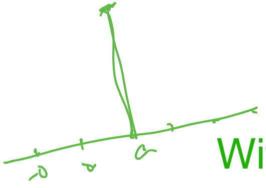
Q: What's the action distribution induced by ϵ-greedy at iteration t?

Practical Consideration

Instead of setting a hard threshold for explore and commit, we often interleave explore and exploitation

ϵ-greedy:

Every iteration t:

$\delta(a) =$ 

With probability $1 - \epsilon$, we play $a_t = \pi^t(x_t)$,

and w/ probability ϵ, we play $a_t \sim \text{Unif}(\mathscr{A})$

Q: What's the action distribution induced by ϵ-greedy at iteration t?

$$a \sim p_t, \quad p_t = (1 - \epsilon)\delta(\pi^t(x_t)) + \epsilon \text{Unif}(\mathscr{A})$$

$\epsilon \to 0$ Exploitation

$\epsilon \to 1$ uniform explore

Put things together: $\epsilon-$greedy

For $t = 0 \to \infty$ (# interleve exploration & exploitation)

1. Observe $x_t \sim \mu$

Put things together: $\epsilon-$greedy

For $t = 0 \rightarrow \infty$ (# interleve exploration & exploitation)

1. Observe $x_t \sim \mu$

2. Use ϵ-greedy to form action distribution $p_t = (1 - \epsilon)\delta(\pi^t(x_t)) + \epsilon\text{Unif}(\mathscr{A})$

Put things together: $\epsilon-$greedy

For $t = 0 \rightarrow \infty$ (# interleve exploration & exploitation)

1. Observe $x_t \sim \mu$

2. Use ϵ-greedy to form action distribution $p_t = (1 - \epsilon)\delta(\pi^t(x_t)) + \epsilon\text{Unif}(\mathscr{A})$

3. Play $a_t \sim p_t$, and observe $r_t := r(x_t, a_t)$

Put things together: $\epsilon-$greedy

For $t = 0 \to \infty$ (# interleve exploration & exploitation)

1. Observe $x_t \sim \mu$

2. Use ϵ-greedy to form action distribution $p_t = (1-\epsilon)\delta(\pi^t(x_t)) + \epsilon\text{Unif}(\mathscr{A})$

3. Play $a_t \sim p_t$, and observe $r_t := r(x_t, a_t)$

4. Via IW, form unbiased estimate $\widehat{\mathbf{r}}_t$

$$\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ r_t / p_t(a_t) \\ 0 \\ \vdots \end{bmatrix}$$

Put things together: ϵ—greedy

For $t = 0 \to \infty$ (# interleve exploration & exploitation)

1. Observe $x_t \sim \mu$
2. Use ϵ-greedy to form action distribution $p_t = (1 - \epsilon)\delta(\pi^t(x_t)) + \epsilon\text{Unif}(\mathscr{A})$
3. Play $a_t \sim p_t$, and observe $r_t := r(x_t, a_t)$
4. Via IW, form unbiased estimate $\widehat{\mathbf{r}}_t$
5. Update via RSC oracle: $\pi^{t+1} = \arg\max_{\pi \in \Pi} \sum_{i=1}^{t} \widehat{\mathbf{r}}_i[\pi(x_i)]$

Put things together: $\epsilon-$greedy

For $t = 0 \to \infty$ (# interleve exploration & exploitation)

1. Observe $x_t \sim \mu$

2. Use ϵ-greedy to form action distribution $p_t = (1-\epsilon)\delta(\pi^t(x_t)) + \epsilon\text{Unif}(\mathscr{A})$

3. Play $a_t \sim p_t$, and observe $r_t := r(x_t, a_t)$

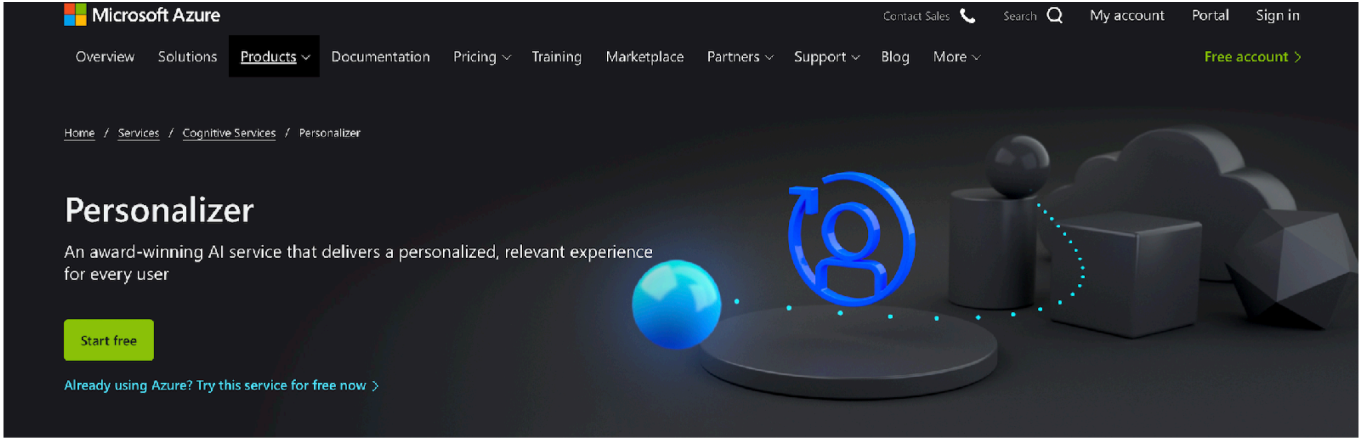
4. Via IW, form unbiased estimate $\mathbf{r}_t$

5. Update via RSC oracle: $\pi^{t+1} = \arg\max_{\pi \in \Pi} \sum_{i=1}^{t} \mathbf{r}_i[\pi(x_i)]$

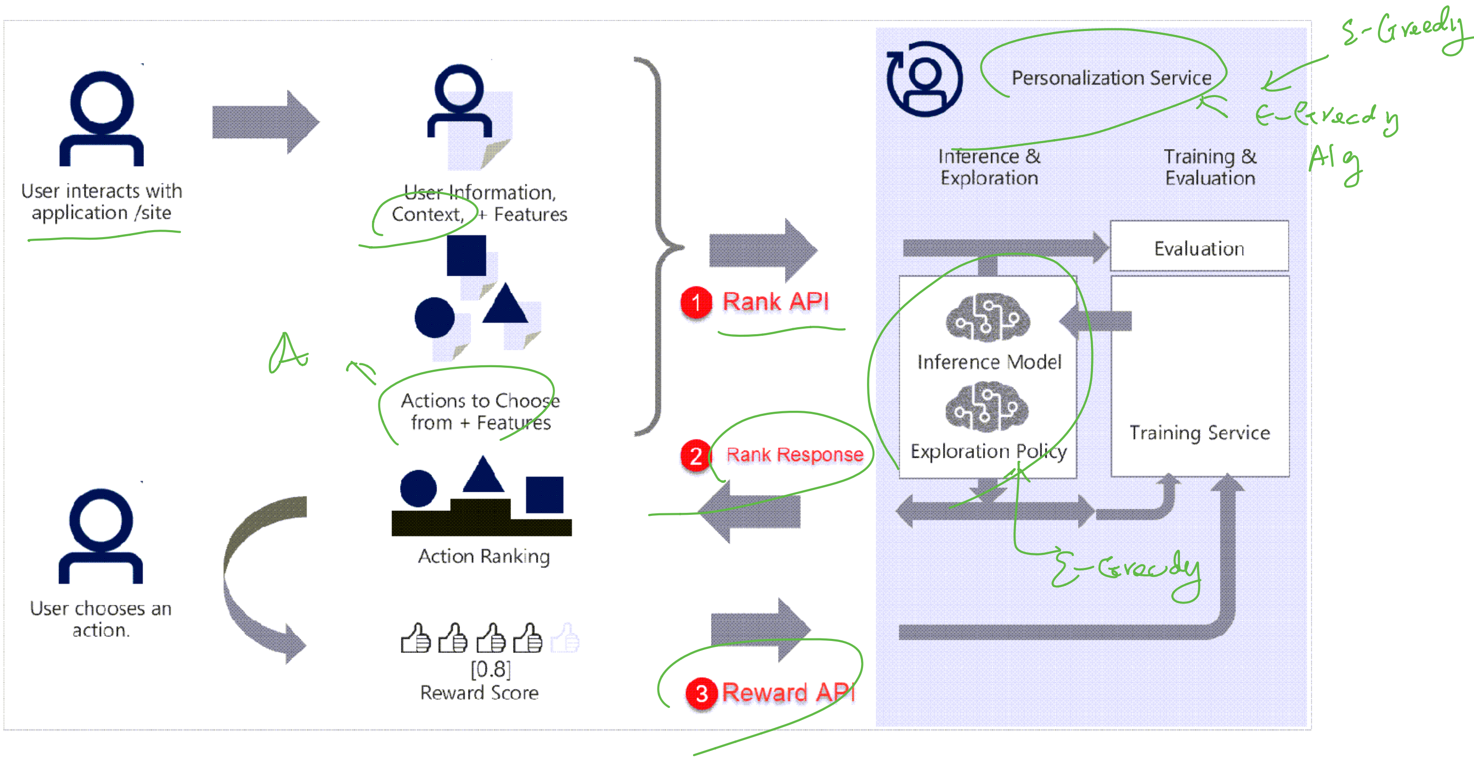
(Additionally 6. Gradually decay ϵ...)

CB algorithm is being used in real world application at Microsoft:

https://azure.microsoft.com/en-us/services/cognitive-services/personalizer/



Framework


User interacts with application /site
User Information, Context, + Features
Actions to Choose from + Features
A
Action Ranking
User chooses an action.
[0.8]
Reward Score
1 Rank API
2 Rank Response
3 Reward API
Personalization Service
ε-Greedy
ε-Greedy Alg
Inference & Exploration
Training & Evaluation
Evaluation
Inference Model
Exploration Policy
Training Service
ε-Greedy