Multi Armed Bandits

Sandeep Juneja

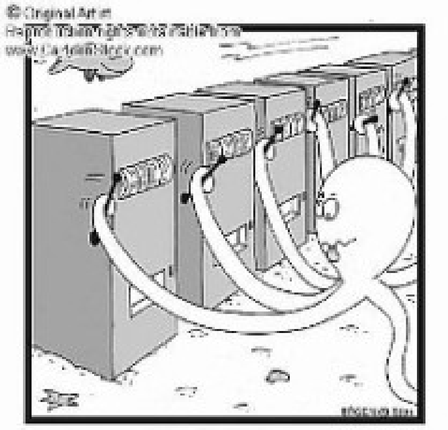
Tata Institute
Mumbai, India

CoRe, IGIDR

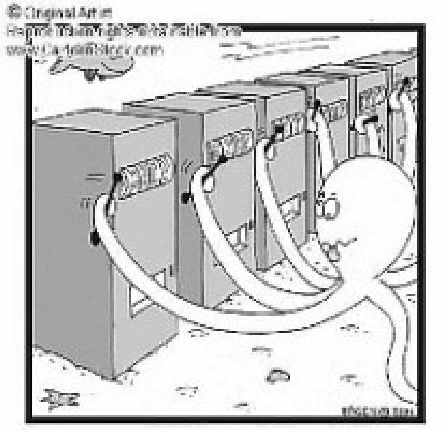
November 05, 2019

The Classical Multi-Armed Bandit Problem

The Classical Multi-Armed Bandit Problem



The Classical Multi-Armed Bandit Problem



‘in the long run they are as effective as human bandits in separating the victim from his money.’ (Lai and Robbins 1985)

Two Arm Bandit Problem

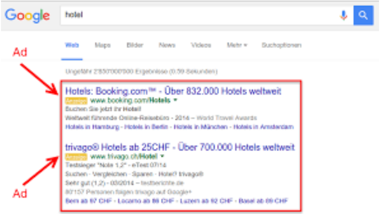
- Two bandits
  - For arm 1: Success probability is p_1
  - For arm 2: Success probability is p_2
- Sequentially play one arm at each trial $t = 1, 2, 3, \ldots$
- Each time reward r_t is obtained - r_t equals 1 with probability p_i, 0 otherwise
  - Want to play arm with larger p_i.
  - But we do not know p_1 or p_2. How to proceed?
  - Explore and exploit. Bandit setting clarifies the fundamental tradeoff between explore and exploit.

Clinical trials

- Five experimental drugs. Which drug to give patients?

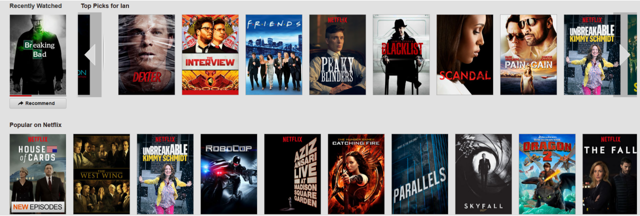
  'it seems apparent that a considerable saving of individuals otherwise sacrificed to the inferior (drug) treatment might be effected' Thompson, 1933

Placing advertisements on Google



- *When a visitor clicks on a display advertisement on a member website, a portion of the revenue is paid to the site owner while Google keeps part of the fee. Due to the breadth of companies advertising through the network, entire businesses depend on AdSense as their primary source of income.* Bulk of 110.8 Billion Google revenue in 2017.
- Which advts. to place to maximise clicks?

Recommendation Systems



- Which type of movie to recommend to a customer? Drama, comedy, western classic.
- Which movies to place to maximize viewer selection?

Some other applications

- Which trading strategy gives the best risk adjusted returns

Some other applications

- Which trading strategy gives the best risk adjusted returns

- In transportation, which route to take among many

Stochastic Multi Armed Bandit Problem

- Multi armed bandit framework provides perhaps the simplest setting for sequential learning and decision making

Stochastic Multi Armed Bandit Problem

- Multi armed bandit framework provides perhaps the simplest setting for sequential learning and decision making
- It crisply demonstrates the explore versus exploit trade-off

Stochastic Multi Armed Bandit Problem

- Multi armed bandit framework provides perhaps the simplest setting for sequential learning and decision making
- It crisply demonstrates the explore versus exploit trade-off
- We consider

Stochastic Multi Armed Bandit Problem

- Multi armed bandit framework provides perhaps the simplest setting for sequential learning and decision making
- It crisply demonstrates the explore versus exploit trade-off
- We consider
  - Regret minimization: Pull arms (i.e., sample from probability distributions) to maximize expected reward, or equivalently, minimise expected regret

Stochastic Multi Armed Bandit Problem

- Multi armed bandit framework provides perhaps the simplest setting for sequential learning and decision making
- It crisply demonstrates the explore versus exploit trade-off
- We consider
  - Regret minimization: Pull arms (i.e., sample from probability distributions) to maximize expected reward, or equivalently, minimise expected regret
  - Pure exploration to find the best arm.

Stochastic Multi Armed Bandit Problem

- Multi armed bandit framework provides perhaps the simplest setting for sequential learning and decision making
- It crisply demonstrates the explore versus exploit trade-off
- We consider
  - Regret minimization: Pull arms (i.e., sample from probability distributions) to maximize expected reward, or equivalently, minimise expected regret
  - Pure exploration to find the best arm.
  - How many questions to ask in an interview? How hard should these questions be?

Regret minimization

- Given K arms, each arm when pulled gives a random reward. The reward vector at time t is given by

$$X_t = (X_{1,t}, X_{2,t}, \ldots, X_{K,t})$$

where mean $EX_{i,t} = \mu_i$.

Regret minimization

- Given K arms, each arm when pulled gives a random reward. The reward vector at time t is given by

$$X_t = (X_{1,t}, X_{2,t}, \ldots, X_{K,t})$$

where mean $EX_{i,t} = \mu_i$.

- Rewards $X_{i,t}$ from each arm are iid and lie within $[0, 1]$.

Regret minimization

- Given K arms, each arm when pulled gives a random reward. The reward vector at time t is given by

$$X_t = (X_{1,t}, X_{2,t}, \ldots, X_{K,t})$$

where mean $EX_{i,t} = \mu_i$.

- Rewards $X_{i,t}$ from each arm are iid and lie within $[0, 1]$.

- At each iteration $t =, 1, 2, 3, \ldots$ learner/algorithm pulls a single arm J_t and receives a reward $X_{J_t,t}$. The learner does not observe the rewards from other arms

- The regret equals

$$R(n) = \sum_{t=1}^{n} X_{j^*,t} - \sum_{t=1}^{n} X_{J_t,t}$$

where j^* is the index of the best arm.

- The regret equals

$$R(n) = \sum_{t=1}^{n} X_{j^*,t} - \sum_{t=1}^{n} X_{J_t,t}$$

where j^* is the index of the best arm.

- The aim of regret minimization is to sequentially pull arms so as to minimise the expected regret

$$ER(n) = n \times \mu^* - \sum_{t=1}^{n} EX_{J_t,t} = \sum_{i=1}^{K} ET_i(n) \times \Delta_i$$

where $T_i(n)$ denote the number of times arm i pulled in n trials. $\Delta_i = \mu^* - \mu_i$. $\mu^* = \max_{i \le K} EX_{i,t}$

Equal sample strategy

- Each arm is given equal number of samples n/K.

Equal sample strategy

- Each arm is given equal number of samples n/K.
- Regret equals
$$\left(\frac{1}{K}\sum_a \Delta_a\right) n$$

Equal sample strategy

- Each arm is given equal number of samples n/K.
- Regret equals
$$\left(\frac{1}{K}\sum_a \Delta_a\right) n$$
- It is linear in n

Greedy strategy

- $J_{t+1} = \arg\max_a \hat{\mu}_a(t)$, where

Greedy strategy

- $J_{t+1} = \arg\max_a \hat{\mu}_a(t)$, where

$$\hat{\mu}_a(t) = \frac{1}{T_a(t)} \sum_{i=1}^{t} X_{a,i} \times I(J_i = a)$$

- Regret is linearly bounded from below by

$$(\mu_1 - \mu_2) \times (1 - \mu_1) \times \mu_2 \times n$$

when μ_1 corresponds to the largest mean, and μ_2 to second largest.

Greedy strategy

- $J_{t+1} = \arg\max_a \hat{\mu}_a(t)$, where

$$\hat{\mu}_a(t) = \frac{1}{T_a(t)} \sum_{i=1}^{t} X_{a,i} \times I(J_i = a)$$

- Regret is linearly bounded from below by

$$(\mu_1 - \mu_2) \times (1 - \mu_1) \times \mu_2 \times n$$

  when μ_1 corresponds to the largest mean, and μ_2 to second largest.

- Is sub-linear regret achievable?

Explore then commit strategy

- Set m to be $\leq n/K$ (n is sampling budget, K is number of players

Explore then commit strategy

- Set m to be $\leq n/K$ (n is sampling budget, K is number of players
- Sample each arm m times. Thereafter, sample the observed best arm

$$\hat{a} = \arg\max_a \hat{\mu}(Km)$$

for remaining $n - Km$ trials.

Explore then commit strategy

- Set m to be $\leq n/K$ (n is sampling budget, K is number of players
- Sample each arm m times. Thereafter, sample the observed best arm
$$\hat{a} = \arg\max_a \hat{\mu}(Km)$$
for remaining $n - Km$ trials.
- Regret in two arms $N(\mu_1, 1)$ and $N(\mu_2, 1)$ setting. $\Delta = \mu_1 - \mu_2 > 0$.
$$m\Delta + (n - 2m)EI(\hat{a} = 2) \leq m\Delta + n \times \exp(-m\Delta^2/4)$$

Explore then commit strategy

- Regret in two arms $N(\mu_1, 1)$ and $N(\mu_2, 1)$ setting. $\Delta = \mu_1 - \mu_2 > 0$.

$$m\Delta + (n - 2m)EI(\hat{a} = 2) \leq m\Delta + n \times \exp(-m\Delta^2/4)$$

Explore then commit strategy

- Regret in two arms $N(\mu_1, 1)$ and $N(\mu_2, 1)$ setting.
  $\Delta = \mu_1 - \mu_2 > 0$.

$$m\Delta + (n - 2m)EI(\hat{a} = 2) \leq m\Delta + n \times \exp(-m\Delta^2/4)$$

- RHS minimized at $m = \frac{4}{\Delta^2} \log\left(\frac{n\Delta}{4}\right)$

Explore then commit strategy

- Regret in two arms $N(\mu_1, 1)$ and $N(\mu_2, 1)$ setting. $\Delta = \mu_1 - \mu_2 > 0$.

$$m\Delta + (n-2m)EI(\hat{a}=2) \leq m\Delta + n \times \exp(-m\Delta^2/4)$$

- RHS minimized at $m = \frac{4}{\Delta^2} \log\left(\frac{n\Delta}{4}\right)$

- Regret $\leq \frac{4}{\Delta}\left(\log\left(\frac{n\Delta}{4}\right) + 1\right)$

Explore then commit strategy

- Regret in two arms $N(\mu_1, 1)$ and $N(\mu_2, 1)$ setting. $\Delta = \mu_1 - \mu_2 > 0$.

$$m\Delta + (n - 2m)EI(\hat{a} = 2) \leq m\Delta + n \times \exp(-m\Delta^2/4)$$

- RHS minimized at $m = \frac{4}{\Delta^2} \log\left(\frac{n\Delta}{4}\right)$
- Regret $\leq \frac{4}{\Delta}\left(\log\left(\frac{n\Delta}{4}\right) + 1\right)$
- Logarithmic regret! Requires knowledge of n and Δ.

Explore then commit strategy: Random switch time

- Two arms $N(\mu_1, 1)$ and $N(\mu_2, 1)$ setting. $\Delta = \mu_1 - \mu_2 > 0$.

Explore then commit strategy: Random switch time

- Two arms $N(\mu_1, 1)$ and $N(\mu_2, 1)$ setting. $\Delta = \mu_1 - \mu_2 > 0$.
- Explore uniformly till a random time

$$\tau = \inf\left[t : |\hat{\mu}_1(t) - \hat{\mu}_2(t)| \geq \left(\frac{8\log(n/t)}{t}\right)^{1/2}\right]$$

Explore then commit strategy: Random switch time

- Two arms $N(\mu_1, 1)$ and $N(\mu_2, 1)$ setting. $\Delta = \mu_1 - \mu_2 > 0$.

- Explore uniformly till a random time

$$\tau = \inf \left[t : |\hat{\mu}_1(t) - \hat{\mu}_2(t)| \geq \left(\frac{8 \log(n/t)}{t} \right)^{1/2} \right]$$

- Sample the best observed arm thereafter

Explore then commit strategy: Random switch time

- Two arms $N(\mu_1, 1)$ and $N(\mu_2, 1)$ setting. $\Delta = \mu_1 - \mu_2 > 0$.
- Explore uniformly till a random time

$$\tau = \inf \left[t : |\hat{\mu}_1(t) - \hat{\mu}_2(t)| \geq \left(\frac{8 \log(n/t)}{t} \right)^{1/2} \right]$$

- Sample the best observed arm thereafter
- Regret $\leq \frac{4}{\Delta} \log n\Delta + C(\log n)^{1/2}$

Explore then commit strategy: Random switch time

- Two arms $N(\mu_1, 1)$ and $N(\mu_2, 1)$ setting. $\Delta = \mu_1 - \mu_2 > 0$.
- Explore uniformly till a random time

$$\tau = \inf\left[t : |\hat{\mu}_1(t) - \hat{\mu}_2(t)| \geq \left(\frac{8\log(n/t)}{t}\right)^{1/2}\right]$$

- Sample the best observed arm thereafter
- Regret $\leq \frac{4}{\Delta}\log n\Delta + C(\log n)^{1/2}$
- No need to know Δ.

The ϵ Greedy Policy Auer, Cesa-Bianchi,Fisher 2002

- At step t, sample each arm uniformly with probability ϵ_t.

The ϵ Greedy Policy Auer, Cesa-Bianchi,Fisher 2002

- At step t, sample each arm uniformly with probability ϵ_t.
- Sample the arm with best sample mean so far with probability $1 - \epsilon_t$

The ϵ Greedy Policy Auer, Cesa-Bianchi,Fisher 2002

- At step t, sample each arm uniformly with probability ϵ_t.
- Sample the arm with best sample mean so far with probability $1 - \epsilon_t$
- Choose

$$\epsilon_t = \min\left(\frac{C}{t}, 1\right)$$

for C a sufficiently large constant

- For n large enough

$$ET_i(n) \leq \frac{D}{\Delta^2} \log n$$

for a constant $D > 0$ (here $\Delta = \min_{i \leq K} \Delta_i$)

- For n large enough

$$ET_i(n) \leq \frac{D}{\Delta^2} \log n$$

for a constant $D > 0$ (here $\Delta = \min_{i \leq K} \Delta_i$)

- The regret

$$ER(n) \leq D \sum_{i=1}^{K} \frac{\Delta_i}{\Delta^2} \log n$$

- For n large enough

$$ET_i(n) \leq \frac{D}{\Delta^2} \log n$$

for a constant $D > 0$ (here $\Delta = \min_{i \leq K} \Delta_i$)

- The regret

$$ER(n) \leq D \sum_{i=1}^{K} \frac{\Delta_i}{\Delta^2} \log n$$

- Logarithmic regret!

Upper Confidence Bound (UCB) Algorithm

- Form an optimistic upper confidence bound (UCB) on each arm

Upper Confidence Bound (UCB) Algorithm

- Form an optimistic upper confidence bound (UCB) on each arm
- This UCB is greater than the sample average but converges to it as the number of samples increase

Upper Confidence Bound (UCB) Algorithm

- Form an optimistic upper confidence bound (UCB) on each arm
- This UCB is greater than the sample average but converges to it as the number of samples increase
- It increases if arm is not sampled for a long time - encouraging exploration

Upper Confidence Bound (UCB) Algorithm

- Form an optimistic upper confidence bound (UCB) on each arm
- This UCB is greater than the sample average but converges to it as the number of samples increase
- It increases if arm is not sampled for a long time - encouraging exploration
- Algorithm simply involves sampling the arm with the largest UCB

Example of Upper Confidence Bound (UCB) Algorithm

UCB Algorithm

1. Sample each arm once

Example of Upper Confidence Bound (UCB) Algorithm

UCB Algorithm

1. Sample each arm once
2. At each step t select an arm with index I_t chosen as

$$I_t = \arg \max_{i=1,\ldots,K} \left(\hat{X}_{i,T_i(t-1)} + \sqrt{\frac{2 \log t}{T_i(t-1)}} \right)$$

Example of Upper Confidence Bound (UCB) Algorithm

UCB Algorithm

1. Sample each arm once
2. At each step t select an arm with index I_t chosen as

$$I_t = \arg\max_{i=1,\ldots,K} \left(\hat{X}_{i,T_i(t-1)} + \sqrt{\frac{2\log t}{T_i(t-1)}} \right)$$

- UCB does a good trade-off between explore and exploit. Each sub-optimal arm is sampled $O(\log n)$ number of times in n steps.

Example of Upper Confidence Bound (UCB) Algorithm

UCB Algorithm

1. Sample each arm once
2. At each step t select an arm with index I_t chosen as

$$I_t = \arg\max_{i=1,\dots,K} \left(\hat{X}_{i,T_i(t-1)} + \sqrt{\frac{2\log t}{T_i(t-1)}} \right)$$

- UCB does a good trade-off between explore and exploit. Each sub-optimal arm is sampled $O(\log n)$ number of times in n steps.
- The expected regret is of $O(\log n)$ in n steps. Better than ϵ greedy

- Can show that

$$ET_i(n) \leq \frac{8 \log n}{\Delta_i^2} + 1 + \frac{\pi^2}{3}.$$

- The regret therefore is also of order $\log n$.

Regret Minimization: Many variants, generalizations exist

- Adversarial bandits

Regret Minimization: Many variants, generalizations exist

- Adversarial bandits
- Contextual bandits

Regret Minimization: Many variants, generalizations exist

- Adversarial bandits
- Contextual bandits
- Continuous arms

Regret Minimization: Many variants, generalizations exist

- Adversarial bandits
- Contextual bandits
- Continuous arms
- General arm distributions

Regret Minimization: Many variants, generalizations exist

- Adversarial bandits
- Contextual bandits
- Continuous arms
- General arm distributions
- Lower bounds on regret for stochastic bandits are known (Lai and Robbins 85)

$$ER(n) \geq \sum_{i \neq i^*} \frac{1}{KL(F_i || F^*)} \log n$$

Regret Minimization: Many variants, generalizations exist

- Adversarial bandits
- Contextual bandits
- Continuous arms
- General arm distributions
- Lower bounds on regret for stochastic bandits are known (Lai and Robbins 85)

$$ER(n) \geq \sum_{i \neq i^*} \frac{1}{KL(F_i || F^*)} \log n$$

- KL-UCB algorithms that match this for large n have been developed

Best arm pure exploration problems

Selection of the Best Arm

- K different arms or probability distributions are compared.

Selection of the Best Arm

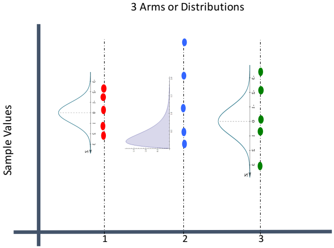
- K different arms or probability distributions are compared.

- Do not know the underlying distributions but can generate samples from them.

Selection of the Best Arm

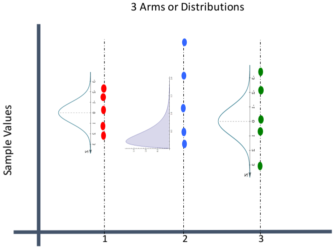
- K different arms or probability distributions are compared.
- Do not know the underlying distributions but can generate samples from them.
- Goal is only to identify the population with the largest mean and not to actually estimate the means.

Classical Monte Carlo problem: Finding a distribution or *arm* with the largest mean



- Do not know the underlying K distributions but can generate samples from them

Classical Monte Carlo problem: Finding a distribution or *arm* with the largest mean



- Do not know the underlying K distributions but can generate samples from them
- Through sequential sampling, identify the population with the largest mean with probability of error $\leq$ pre-specified δ

Some applications

- Given stochastic models of different road network designs, finding the one with least average congestion.

Some applications

- Given stochastic models of different road network designs, finding the one with least average congestion.

- Given a manufacturing system evaluating the best maintenance strategy.

Some applications

- Given stochastic models of different road network designs, finding the one with least average congestion.
- Given a manufacturing system evaluating the best maintenance strategy.
- Given many medicinal treatments for a given disease, finding the one that causes maximum benefit on average.

Brief Literature Review

- Statistics: Bechhofer et. al. (1968) Uniform sampling, Paulson (1964) - elimination based. Earlier Chernoff (1959), Albert (1961)

Brief Literature Review

- Statistics: Bechhofer et. al. (1968) Uniform sampling, Paulson (1964) - elimination based. Earlier Chernoff (1959), Albert (1961)
- Simulation: Bechhofer, Goldsman, Nelson and others 90's, 2000's, Ho et. al. (1990), Dai (1996), Chen et al (2000), Glynn and J (2004)

Brief Literature Review

- Statistics: Bechhofer et. al. (1968) Uniform sampling, Paulson (1964) - elimination based. Earlier Chernoff (1959), Albert (1961)
- Simulation: Bechhofer, Goldsman, Nelson and others 90's, 2000's, Ho et. al. (1990), Dai (1996), Chen et al (2000), Glynn and J (2004)
- Computer Science - Evan-Dar et. al. (2006), Bubeck, Audibert (2010), Kaufmann, Cappe, Garivier (2016), Garivier, Kaufmann (2016), Russo (2016)

Popular Successive Rejection Algorithm

Successive rejection algorithm

- Total K arms. Each arm a when sampled gives a Bounded reward in $[0, 1]$ with mean μ_a.

Successive rejection algorithm

- Total K arms. Each arm a when sampled gives a Bounded reward in $[0, 1]$ with mean μ_a.
- Let $a^* = \arg\max_{a \in A} \mu_a$ and let $\Delta_a = \mu_{a^*} - \mu_a$.

Successive rejection algorithm

- Total K arms. Each arm a when sampled gives a Bounded reward in $[0, 1]$ with mean μ_a.
- Let $a^* = \arg\max_{a \in A} \mu_a$ and let $\Delta_a = \mu_{a^*} - \mu_a$.
- Even Dar et al. 2006 devise a sequential sampling strategy to find a^* with probability at least $1 - \delta$.

Successive rejection algorithm

- Total K arms. Each arm a when sampled gives a Bounded reward in $[0, 1]$ with mean μ_a.
- Let $a^* = \arg\max_{a \in A} \mu_a$ and let $\Delta_a = \mu_{a^*} - \mu_a$.
- Even Dar et al. 2006 devise a sequential sampling strategy to find a^* with probability at least $1 - \delta$.
- Expected computational effort

$$O\left(\sum_{a \neq a^*} \frac{\log(K/\delta)}{\Delta_a^2}\right).$$

Successive rejection algorithm

- Sample every arm a once and let $\hat{\mu}_a^t$ be the average reward of arm a by time t;

Successive rejection algorithm

- Sample every arm a once and let $\hat{\mu}_a^t$ be the average reward of arm a by time t;
- Each arm a such that

$$\hat{\mu}_{\max}^t - \hat{\mu}_a^t \geq 2\alpha_t$$

is removed from consideration. $\alpha_t = \sqrt{\log(5nt^2/\delta)/(2t)}$;

Successive rejection algorithm

- Sample every arm a once and let $\hat{\mu}_a^t$ be the average reward of arm a by time t;
- Each arm a such that

$$\hat{\mu}_{\text{max}}^t - \hat{\mu}_a^t \geq 2\alpha_t$$

is removed from consideration. $\alpha_t = \sqrt{\log(5nt^2/\delta)/(2t)}$;

- $t = t + 1$; Repeat till one arm left.

Proof requires Hoeffding's Inequality

- Suppose that $Y_1, Y_2, \ldots, Y_n$ are independent identically distributed random variables taking values in $[0, 1]$.
- Let

$$S_n = Y_1 + Y_2 + \ldots + Y_n$$

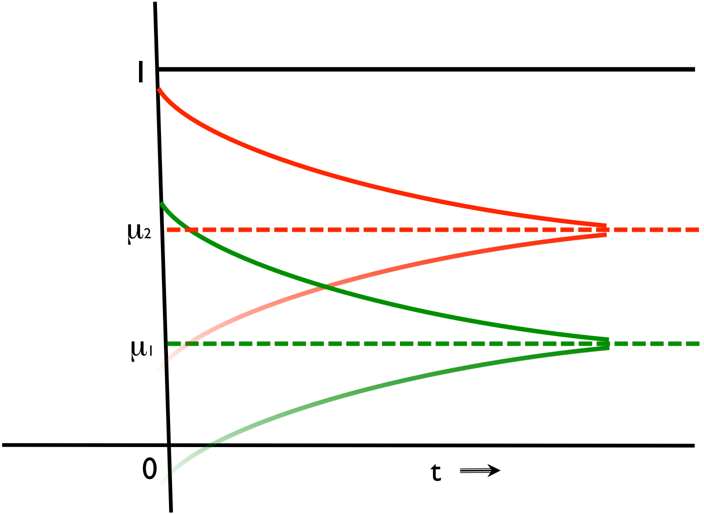
and $\mu = EY_i$. Then, for all $a \geq 0$,

$$P(\frac{S_n}{n} \geq \mu + a) \leq \exp(-2na^2)$$

and

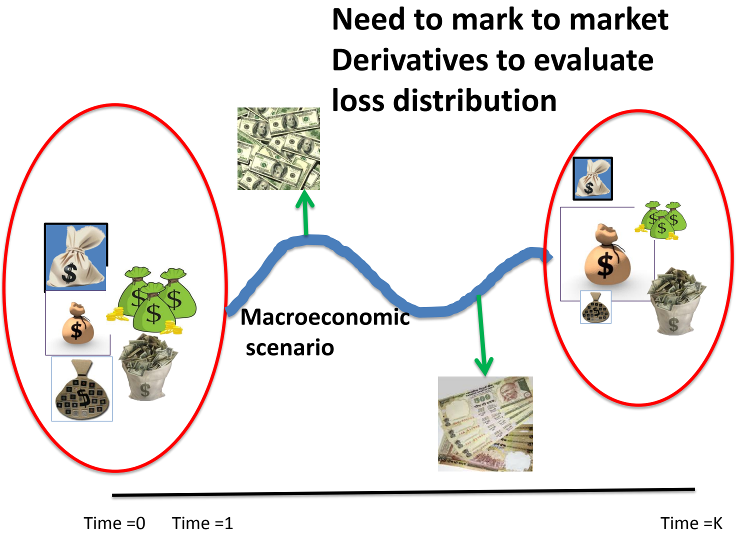
$$P(\frac{S_n}{n} \leq \mu - a) \leq \exp(-2na^2)$$

Key idea

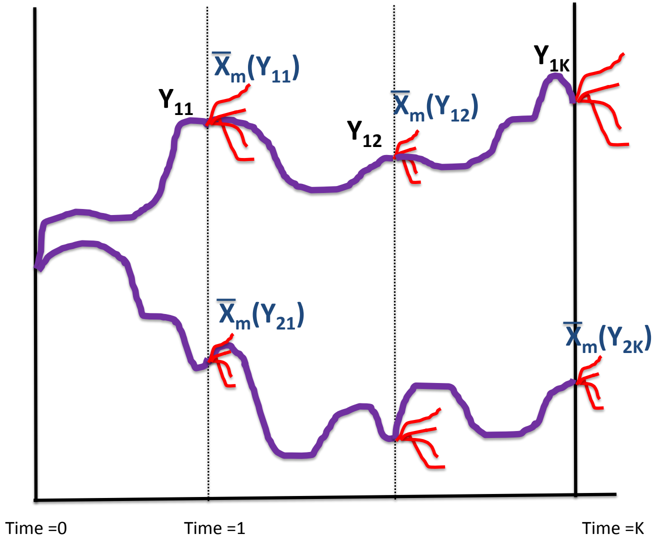


Nested simulation in finance

Estimating $P\left(\max_{t=1,\ldots,K}(EX_t|Y_t) > u\right)$



Naive estimator $\frac{1}{n}\sum_{i=1}^{n} I_i(\max_{t=1,\ldots,K} \bar{X}_m(Y_{i,t}) > u)$



Abstract Partition Identification Problem

We consider: Finding the correct partition for vector of distributions

- Ω is a collection of vectors $\omega = (\nu_1, \ldots, \nu_K)$ where each ν_i is a probability distribution.

We consider: Finding the correct partition for vector of distributions

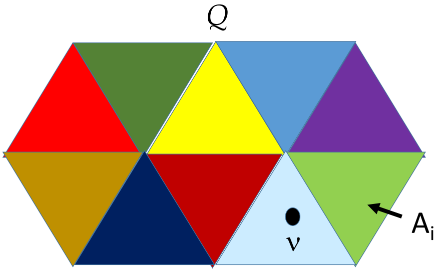
- Ω is a collection of vectors $\omega = (\nu_1, \dots, \nu_K)$ where each ν_i is a probability distribution.
- Can express $\Omega = \cup_{i=1}^{p} A_i$ where the A_i are disjoint

We consider: Finding the correct partition for vector of distributions

- Ω is a collection of vectors $\omega = (\nu_1, \ldots, \nu_K)$ where each ν_i is a probability distribution.
- Can express $\Omega = \cup_{i=1}^{p} A_i$ where the A_i are disjoint
- Given a $\omega \in \Omega$ need to determine which A_i it belongs to.

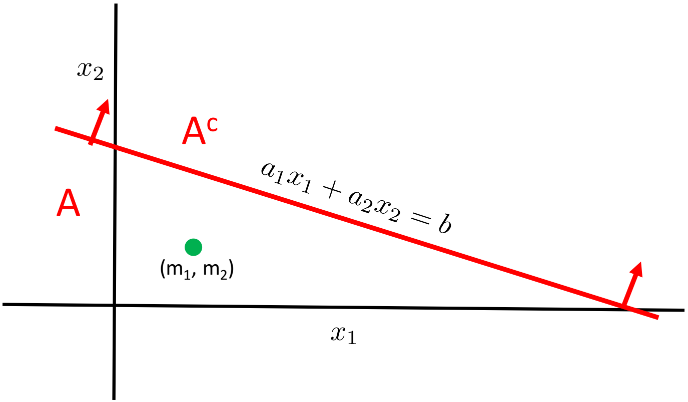
We consider: Finding the correct partition for vector of distributions

- Ω is a collection of vectors $\omega = (\nu_1, \ldots, \nu_K)$ where each ν_i is a probability distribution.
- Can express $\Omega = \cup_{i=1}^{p} A_i$ where the A_i are disjoint
- Given a $\omega \in \Omega$ need to determine which A_i it belongs to.
- Can sample independently from each arm

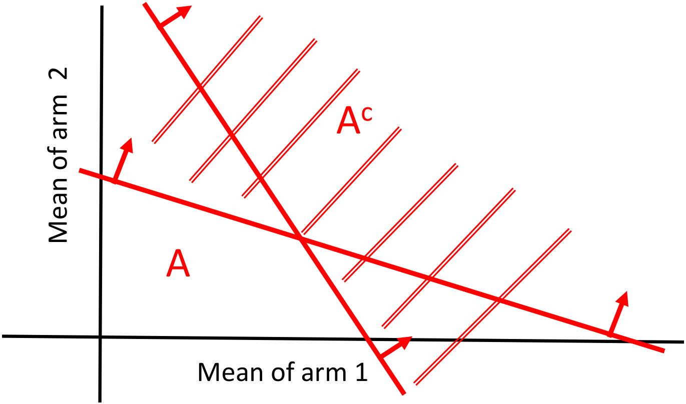


Example: Finding the half-space that contains the vector of means

$\Omega = A \cup A^c$ is a collection of vectors $\omega = (\nu_1, \nu_2)$ where each ν_i is a probability distribution with mean m_i



Finding if the vector of means lies in a convex set, e.g., polytope or its complement



Agenda

- We develop methodology for computing lower bounds on computational effort for δ - correct algorithms.

Agenda

- We develop methodology for computing lower bounds on computational effort for δ - correct algorithms.

- This involves exploiting the geometry of the problem structure; use of duality or minimax theorem.

Agenda

- We develop methodology for computing lower bounds on computational effort for δ - correct algorithms.
- This involves exploiting the geometry of the problem structure; use of duality or minimax theorem.
- We develop δ correct algorithms with matching computational bounds in general settings including the half space problem, the convex and the complement of convex set.

δ-Correct Algorithm

- Given a vector of arms or probability distributions, an algorithm specifies
  - an adaptive sampling strategy
  - a stopping time τ, and finally
  - a recommendation (a subset from the partition)

δ-Correct Algorithm

- Given a vector of arms or probability distributions, an algorithm specifies
  - an adaptive sampling strategy
  - a stopping time τ, and finally
  - a recommendation (a subset from the partition)
- *An algorithm is said to be δ-Correct ,*

δ-Correct Algorithm

- Given a vector of arms or probability distributions, an algorithm specifies
  - an adaptive sampling strategy
  - a stopping time τ, and finally
  - a recommendation (a subset from the partition)
- *An algorithm is said to be δ-Correct ,*
  - if for any $\mu = (\mu_1, \mu_2, \ldots, \mu_K) \in \Omega$,
  - it announces in finite time τ, that μ belongs to some set A_j
  - with the probability of error bounded above by δ, for all $\delta > 0$.

Lower bound relies on a key Inequality

- Relies on change of measure arguments that go back at least to Lai and Robbins 1985.

Lower bound relies on a key Inequality

- Relies on change of measure arguments that go back at least to Lai and Robbins 1985.

- Under δ-correct algorithm (Kauffman, Cappe, Garivier 2016), for

$$\mu = (\mu_1, \mu_2, \ldots, \mu_K) \in A_1$$

and

$$\nu = (\nu_1, \nu_2, \ldots, \nu_K) \in A_1^c$$

where each arm i is pulled N_i times,

Lower bound relies on a key Inequality

- Relies on change of measure arguments that go back at least to Lai and Robbins 1985.
- Under δ-correct algorithm (Kauffman, Cappe, Garivier 2016), for

$$\mu = (\mu_1, \mu_2, \ldots, \mu_K) \in A_1$$

and

$$\nu = (\nu_1, \nu_2, \ldots, \nu_K) \in A_1^c$$

where each arm i is pulled N_i times,

- we have the 'separation cost' inequality

$$\boxed{\sum_{i=1}^{K} E_\mu N_i \times KL(\mu_i || \nu_i) \geq \log\left(\frac{1}{\delta}\right)}$$

Rationale for the lower bound

- If $\mathbf{X} = (X_{i,j} : i \leq K, j \leq N_j)$ denotes the adaptively generated samples by δ-correct algorithm,

Rationale for the lower bound

- If $\mathbf{X} = (X_{i,j} : i \leq K, j \leq N_j)$ denotes the adaptively generated samples by δ-correct algorithm,

$$P_\mu(\mathbf{X} \to A_1) \geq 1 - \delta \quad \text{and, for } \nu \in A_1^c$$

$$P_\nu(\mathbf{X} \to A_1) = E_\mu \exp\left(-\sum_{a=1}^{K}\sum_{j=1}^{N_a} \log \frac{d\mu_a}{d\nu_a}(X_{a,j})\right) I(\mathbf{X} \to A_1) \leq \delta$$

- This leads to the inequality

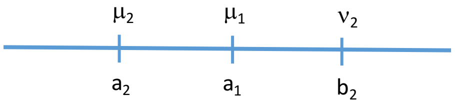
$$\boxed{\sum_{i=1}^{K} E_\mu N_i \times KL(\mu_i || \nu_i) \geq \log\left(\frac{1}{\delta}\right).}$$

Some restrictions necessary on distributions of underlying arms

Selecting the best arm (two arms setting) Glynn and J 2015

Consider $\mu = (\mu_1, \mu_2)$ with means (a_1, a_2) $\quad a_1 > a_2$

and $\quad \nu = (\nu_1, \nu_2)$, $\quad \nu_1 = \mu_1$ with means (a_1, b_2) $\quad b_2 > a_1$



Under δ Correct algorithm lower bound on expected number of samples given to arm 2 under P

$$E_\mu N_2 \, KL(\mu_2 \| \nu_2) >= \log(1/\delta)$$

- Under δ-correct algorithm lower bound on expected number of samples given to arm 2 under P

$$E_{\mu} N_2 \times KL(\mu_2 || \nu_2) \geq \log\left(\frac{1}{\delta}\right).$$

- Under δ-correct algorithm lower bound on expected number of samples given to arm 2 under P

$$E_\mu N_2 \times KL(\mu_2 || \nu_2) \geq \log\left(\frac{1}{\delta}\right).$$

- Then,

$$\boxed{E_\mu N_2 \geq \frac{1}{\inf_{\nu_2 : m_3 > m_1} KL(\mu_2 || \nu_2)} \log\left(\frac{1}{\delta}\right)}$$

- Under δ-correct algorithm lower bound on expected number of samples given to arm 2 under P

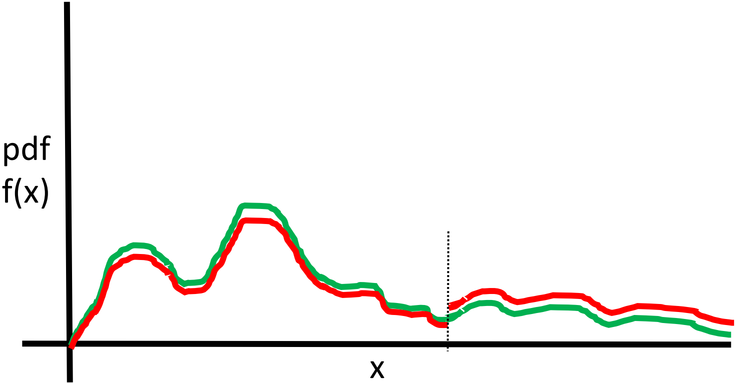
$$E_\mu N_2 \times KL(\mu_2||\nu_2) \geq \log\left(\frac{1}{\delta}\right).$$

- Then,

$$\boxed{E_\mu N_2 \geq \frac{1}{\inf_{\nu_2:m_3>m_1} KL(\mu_2||\nu_2)} \log\left(\frac{1}{\delta}\right)}$$

- Glynn and J. show that if distributions are unbounded, $KL(\mu_2||\nu_2)$ can be made arbitrarily small, hence finite expected time algorithms not feasible without further restrictions

Two dist. - Mean arbitrarily far, KL arbitrarily close



We restrict to one parameter exponential families

- Distribution function of each arm has the form

$$d\mu(\theta, x) = \exp(x\theta - \Lambda(\theta))d\rho(x)$$

 for some constant θ, reference distribution ρ, and appropriate function $\Lambda(\theta)$.

We restrict to one parameter exponential families

- Distribution function of each arm has the form

$$d\mu(\theta, x) = \exp(x\theta - \Lambda(\theta))d\rho(x)$$

 for some constant θ, reference distribution ρ, and appropriate function $\Lambda(\theta)$.

- Examples include Binomial, Poisson, Gaussian with known variance, Gamma distribution with known shape parameter.

We restrict to one parameter exponential families

- Distribution function of each arm has the form

$$d\mu(\theta, x) = \exp(x\theta - \Lambda(\theta))d\rho(x)$$

 for some constant θ, reference distribution ρ, and appropriate function $\Lambda(\theta)$.

- Examples include Binomial, Poisson, Gaussian with known variance, Gamma distribution with known shape parameter.

- This allows us to think of Kullbach Leibler divergence as a function of the means of the distributions.

We restrict to one parameter exponential families

- Distribution function of each arm has the form

$$d\mu(\theta, x) = \exp(x\theta - \Lambda(\theta))d\rho(x)$$

  for some constant θ, reference distribution ρ, and appropriate function $\Lambda(\theta)$.

- Examples include Binomial, Poisson, Gaussian with known variance, Gamma distribution with known shape parameter.

- This allows us to think of Kullbach Leibler divergence as a function of the means of the distributions.

- In the remaining talk, Ω is a collection of vector of parameters in $\Re^K$.

Optimization problem for lower bounds on computational effort

- Recall the key inequality for δ correct algorithms

$$\boxed{\sum_{a=1}^{K} E_\mu N_a \times KL(\mu_a \| \nu_a) \geq \log\left(\frac{1}{\delta}\right)}$$

with

$$\sum_{a=1}^{K} E_\mu N_a = E_\mu \tau$$

- Lower bound on such algorithms, for $\mu \in A_i$,

$$\min \sum_{a=1}^{K} t_a$$

$$\text{s.t.} \quad \inf_{\nu \in A_i^c} \sum_{a=1}^{K} t_a \times KL(\mu_a || \nu_a) \geq 1. \qquad (1)$$

$$t_a \geq 0, \ \forall a.$$

Each t_a needs to scale by $\log(\frac{1}{\delta})$. Re-express (1)

$$\sum_{a=1}^{K} t_a \inf_{\nu \in A_i^c} \sum_{a=1}^{K} \frac{t_a}{\sum_{a=1}^{K} t_a} \times KL(\mu_a || \nu_a) \geq 1,$$

we get an equivalent max-min representation

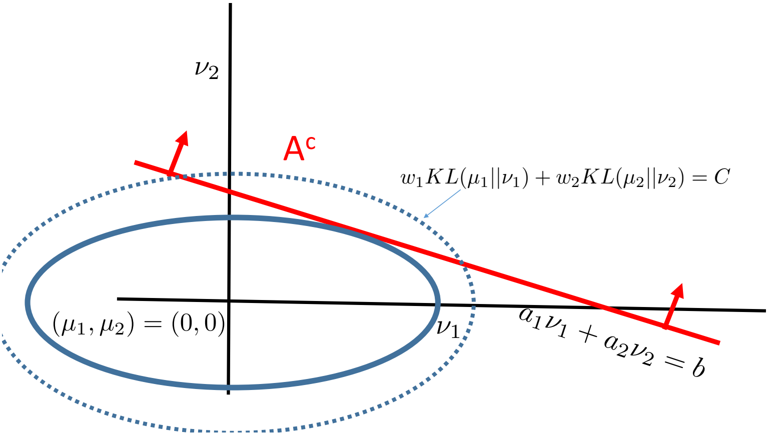
$$\boxed{\max_{\sum_{a=1}^{K} w_a = 1, w_a \geq 0} \inf_{\nu \in A_i^c} \sum_{a=1}^{K} w_a \, KL(\mu_a || \nu_a)}$$

A geometric view when A is a half-space

$$\max_{w_1+w_2=1, w_i \geq 0} \inf_{\nu \in A^c} \left(w_1 \, KL(\mu_1 || \nu_1) + w_2 \, KL(\mu_2 || \nu_2) \right)$$

A geometric view when A is a half-space

$$\max_{w_1+w_2=1, w_i\geq 0} \inf_{\nu\in A^c} \left(w_1\, KL(\mu_1||\nu_1) + w_2\, KL(\mu_2||\nu_2)\right)$$



Characterizing the solution to lower bound

Sets A and A^c are half spaces

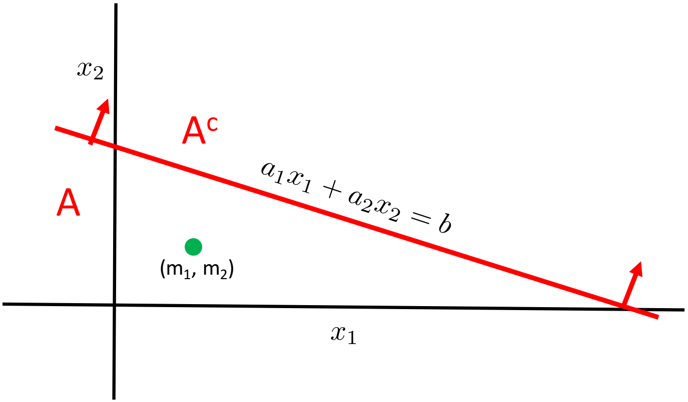
A, a half-space

- Given

$$\mu \in A \triangleq \{\nu \in \Omega : \sum_{i=1}^{K} a_i \nu_i < b\}$$

what restrictions do $\nu \in A^c$ impose on $E_\mu N_a$ for each arm a

μ in a half space



Optimization problem for lower bounds

- $$\max_{\sum_{a=1}^{K} w_a = 1, w_a \geq 0} \inf_{\nu \in A^c} \sum_{a=1}^{K} w_a \, KL(\mu_a || \nu_a)$$

Optimization problem for lower bounds

- $$\max_{\sum_{a=1}^{K} w_a = 1, w_a \geq 0} \inf_{\nu \in A^c} \sum_{a=1}^{K} w_a \, KL(\mu_a || \nu_a)$$

- Using minimax theorem

$$\inf_{\nu \in A^c} \max_{\sum_{a=1}^{K} w_a = 1, w_a \geq 0} \sum_{a=1}^{K} w_a \, KL(\mu_a || \nu_a)$$

Optimization problem for lower bounds

- $$\max_{\sum_{a=1}^K w_a=1, w_a\geq 0} \inf_{\nu\in A^c} \sum_{a=1}^K w_a \, KL(\mu_a||\nu_a)$$

- Using minimax theorem

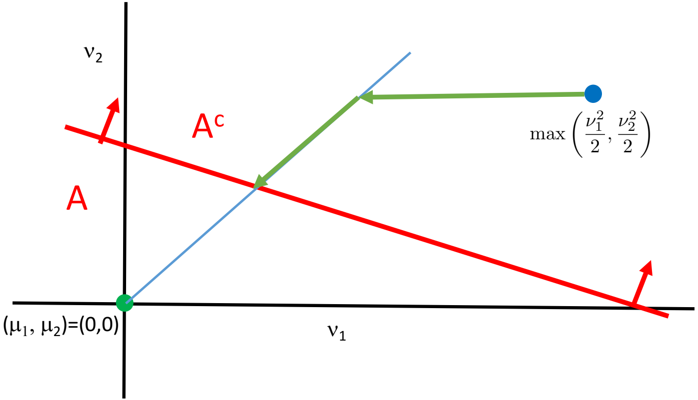
$$\inf_{\nu\in A^c} \max_{\sum_{a=1}^K w_a=1, w_a\geq 0} \sum_{a=1}^K w_a \, KL(\mu_a||\nu_a)$$

- This equals

$$\inf_{\nu\in A^c} \max_{a} KL(\mu_a||\nu_a).$$

Solving $\inf_{\nu \in A^c} \max_a KL(\mu_a || \nu_a)$

- Set $(\mu_1, \mu_2) = (0, 0)$.
- Gaussian distribution with variance 1, so $KL(\mu_i || \nu_i) = \nu_i^2/2$.



- The optimal solution (w^*, ν^*) corresponds to

$$KL(\mu_i || \nu_i^*) = KL(\mu_1 || \nu_1^*) \quad \forall i,$$

$$\sum_{i=1}^{K} a_i \nu_i^* = b.$$

- The optimal solution (w^*, ν^*) corresponds to

$$KL(\mu_i || \nu_i^*) = KL(\mu_1 || \nu_1^*) \quad \forall i,$$

$$\sum_{i=1}^{K} a_i \nu_i^* = b.$$

- The slope matching condition

$$\frac{w_i^*}{a_i} KL'(\mu_i || \nu_i^*) = \frac{w_1^*}{a_1} KL'(\mu_1 || \nu_1^*).$$

- The optimal solution (w^*, ν^*) corresponds to

$$KL(\mu_i || \nu_i^*) = KL(\mu_1 || \nu_1^*) \quad \forall i,$$

$$\sum_{i=1}^{K} a_i \nu_i^* = b.$$

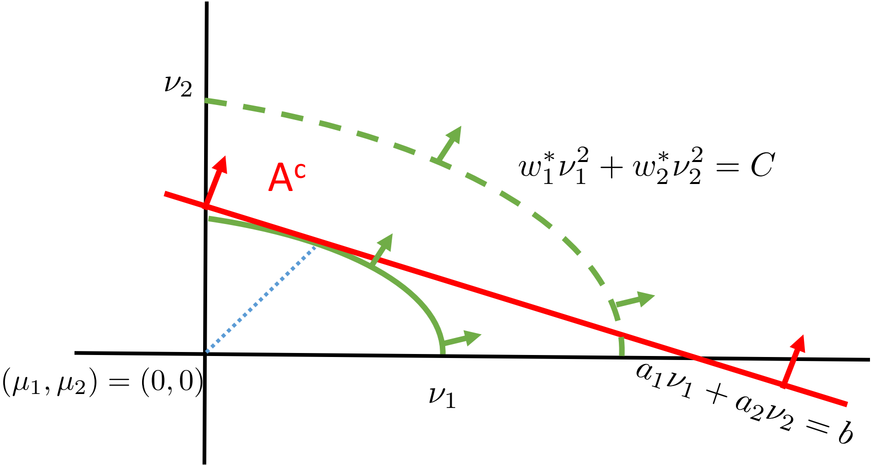
- The slope matching condition

$$\frac{w_i^*}{a_i} KL'(\mu_i || \nu_i^*) = \frac{w_1^*}{a_1} KL'(\mu_1 || \nu_1^*).$$

- And lower bound on expected generated samples

$$KL(\mu_1 || \nu_1^*)^{-1} \times \log(\frac{1}{2.4\delta}).$$

Two Gaussian arms with mean zero



When A^c is convex

- Recall the min-max lower bound problem

$$\max_{\sum_{a=1}^K w_a=1, w_a\geq 0} \inf_{\nu\in A^c} \sum_{a=1}^K w_a\, KL(\mu_a||\nu_a)$$

When A^c is convex

- Recall the min-max lower bound problem

$$\max_{\sum_{a=1}^K w_a=1, w_a \geq 0} \inf_{\nu \in A^c} \sum_{a=1}^{K} w_a \, KL(\mu_a || \nu_a)$$

- **Theorem:** Let (w^*, ν^*) denote an optimal solution.

When A^c is convex

- Recall the min-max lower bound problem

$$\max_{\sum_{a=1}^{K} w_a=1, w_a \geq 0} \inf_{\nu \in A^c} \sum_{a=1}^{K} w_a \, KL(\mu_a || \nu_a)$$

- **Theorem:** Let (w^*, ν^*) denote an optimal solution.
  - ν^* is unique. It solves: $\min_{\nu \in A^c} \max_i K_i(\mu_i | \nu_i)$

When A^c is convex

- Recall the min-max lower bound problem

$$\max_{\sum_{a=1}^K w_a=1, w_a \geq 0} \inf_{\nu \in A^c} \sum_{a=1}^K w_a \, KL(\mu_a || \nu_a)$$

- **Theorem:** Let (w^*, ν^*) denote an optimal solution.
    - ν^* is unique. It solves: $\min_{\nu \in A^c} \max_i K_i(\mu_i | \nu_i)$
    - There exists a maximal $\mathcal{I} \subset \{1, 2, \ldots, K\}$ such that $w_i^* > 0$ for $i \in \mathcal{I}$, $w_i^* = 0$ for rest of i,

When A^c is convex

- Recall the min-max lower bound problem

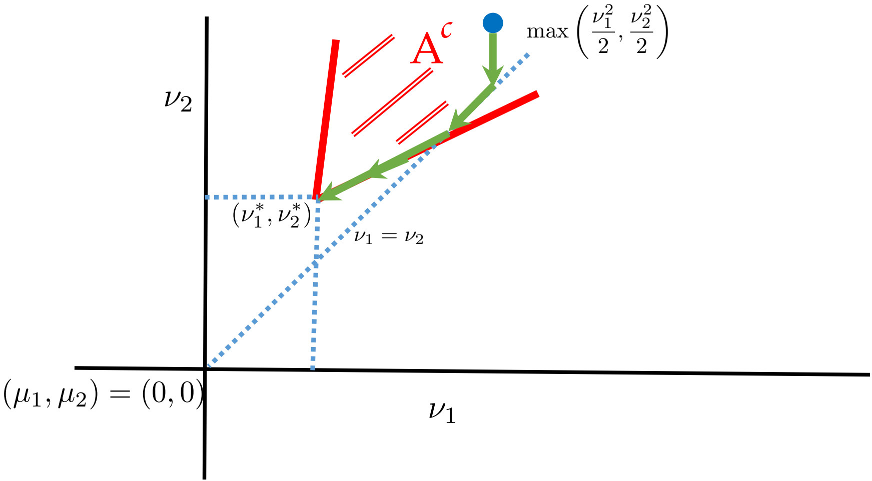
$$\max_{\sum_{a=1}^K w_a=1, w_a\geq 0} \inf_{\nu\in A^c} \sum_{a=1}^{K} w_a\, KL(\mu_a||\nu_a)$$

- **Theorem:** Let (w^*, ν^*) denote an optimal solution.
  - ν^* is unique. It solves: $\min_{\nu\in A^c} \max_i K_i(\mu_i|\nu_i)$
  - There exists a maximal $\mathcal{I} \subset \{1, 2, \ldots, K\}$ such that $w_i^* > 0$ for $i \in \mathcal{I}$, $w_i^* = 0$ for rest of i,

$$KL(\mu_i||\nu_i^*) = Const. \text{ for } i \in \mathcal{I},$$

$$KL(\mu_i|\nu_i^*) < Const \text{ for } i \in \mathcal{I}^c.$$

Optimal soln. when A^c is convex



δ-correct algorithm that matches lower bounds

The tracking algorithm

- Which arm to sample and when to stop

The tracking algorithm

- Which arm to sample and when to stop
- Closely follows Garivier and Kaufmann (2016) that was proposed in the best arm setting

The tracking algorithm

- Which arm to sample and when to stop
- Closely follows Garivier and Kaufmann (2016) that was proposed in the best arm setting
- At sampling step n, ensure that at least about $\sqrt{n}$ samples allocated to each arm.

The tracking algorithm

- Which arm to sample and when to stop
- Closely follows Garivier and Kaufmann (2016) that was proposed in the best arm setting
- At sampling step n, ensure that at least about $\sqrt{n}$ samples allocated to each arm.
- This ensures that with high probability $\hat{\mu}_n$ approximates μ and thus the optimization solution $w^*(\hat{\mu}_n)$ approximates $w^*(\mu)$.

The tracking algorithm

- Which arm to sample and when to stop
- Closely follows Garivier and Kaufmann (2016) that was proposed in the best arm setting
- At sampling step n, ensure that at least about $\sqrt{n}$ samples allocated to each arm.
- This ensures that with high probability $\hat{\mu}_n$ approximates μ and thus the optimization solution $w^*(\hat{\mu}_n)$ approximates $w^*(\mu)$.
- Choose an arm that maximises

$$w^*(\hat{\mu}_n) - \frac{N_i(n)}{n}.$$

Stopping rule motivated by Generalized Likelihood Ratio Method (Chernoff)

- After iteration n, suppose $\hat{\mu}(n) \in \tilde{A}$ (either A or A^c)

- Compute logarithm of

$$\frac{\max_{\mu \in \tilde{A}} \text{ Likelihood Ratio } (\mu)}{\max_{\nu \in \tilde{A}^c} \text{ Likelihood Ratio } (\nu)}.$$

- This equals

$$\inf_{\nu \in \tilde{A}^c} \sum_i \frac{N_i(n)}{n} \times KL(\hat{\mu}_n || \nu_i)$$

Stopping rule

- Let the separation function

$$\beta(n, \delta) = \log\left(\frac{cn}{\delta}\right)$$

for well chosen c.

Stopping rule

- Let the separation function

$$\beta(n, \delta) = \log\left(\frac{cn}{\delta}\right)$$

for well chosen c.

- After iteration n, suppose $\hat{\mu}(n) \in \tilde{A}$ (either A or A^c)

- **If**

$$\inf_{\nu \in \tilde{A}^c} \sum_i \frac{N_i(n)}{n} \times KL(\hat{\mu}_n || \nu_i) \geq \frac{1}{n}\beta(n, \delta)$$

- **then** declare $\mu \in \tilde{A}$

Stopping rule

- Let the separation function

$$\beta(n, \delta) = \log\left(\frac{cn}{\delta}\right)$$

  for well chosen c.

- After iteration n, suppose $\hat{\mu}(n) \in \tilde{A}$ (either A or A^c)

- **If**

$$\inf_{\nu \in \tilde{A}^c} \sum_i \frac{N_i(n)}{n} \times KL(\hat{\mu}_n || \nu_i) \geq \frac{1}{n}\beta(n, \delta)$$

- **then** declare $\mu \in \tilde{A}$

- Else, sample again

Result

Theorem

The algorithm is δ-correct. If $\tau(\delta)$ denotes the stopping time, then

$$\limsup_{\delta \to 0} \frac{E_\mu \tau(\delta)}{\log(1/\delta)} = KL(\mu_1 || \nu_1^*)^{-1}.$$

Perfect interview design

Model of ability and question difficulty

- Suppose candidate's probability of answering a question correctly in an evaluation is

$$\boxed{P(success) = h(p, x)}$$

Model of ability and question difficulty

- Suppose candidate's probability of answering a question correctly in an evaluation is

$$\boxed{P(success) = h(p, x)}$$

- where $p \in \Re^+$ denotes person's ability (unknown) and $x \in \Re^+$ denotes hardness of the question. h increases with p and reduces with x

Model of ability and question difficulty

- Suppose candidate's probability of answering a question correctly in an evaluation is

$$\boxed{P(success) = h(p, x)}$$

- where $p \in \Re^+$ denotes person's ability (unknown) and $x \in \Re^+$ denotes hardness of the question. h increases with p and reduces with x

- Two examples

$$P(success) = \frac{p}{p+x} \text{ or logit model } \frac{1}{1+\exp(\alpha(x-p)+\beta)}.$$

We consider the following ..

- A single candidate has ability p not known to evaluator.

We consider the following ..

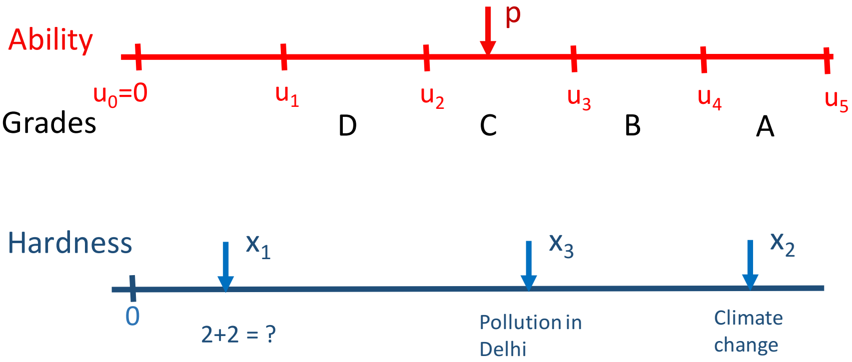
- A single candidate has ability p not known to evaluator.
- Evaluator needs to decide candidate's grade, i.e., the interval $[u_i, u_{i+1})$ in which p lies given

$$0 = u_0 < u_1 < u_2 < \ldots < u_m.$$

- The questions are asked sequentially and adaptively in a pure exploration multi-armed bandit framework

Single candidate, sequential interrogation

Prob. of success = p/(p+x)



Fixed confidence setting

- We develop lower bounds on expected number of questions asked, that hold uniformly for all δ - correct algorithms.

Fixed confidence setting

- We develop lower bounds on expected number of questions asked, that hold uniformly for all δ - correct algorithms.

An algorithm is said to be δ-correct if

- *It asks questions at adaptively chosen levels* $x_1, x_2, \ldots, x_\tau$ *for* $\tau < \infty$,

Fixed confidence setting

- We develop lower bounds on expected number of questions asked, that hold uniformly for all δ - correct algorithms.

An algorithm is said to be δ-correct if

- *It asks questions at adaptively chosen levels $x_1, x_2, \ldots, x_\tau$ for $\tau < \infty$,*
- *It then announces candidate's grade with error probability bounded above by δ, for all $\delta > 0$.*

- We then develop algorithms that up to a dominant term match the lower bound.

- We then develop algorithms that up to a dominant term match the lower bound.

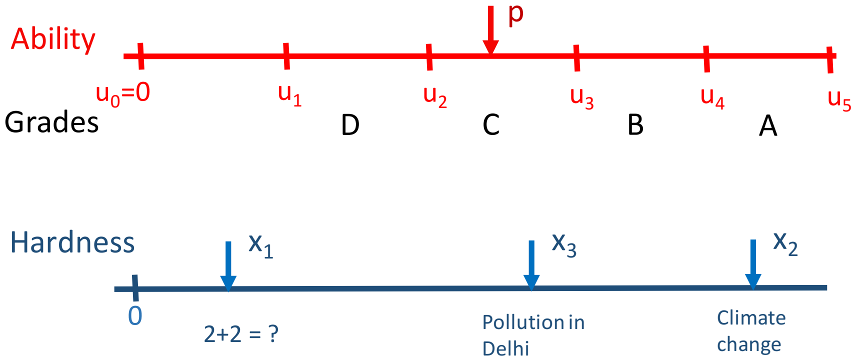
- Key insight is that only up to two level of difficulty questions need to be asked, and in popular settings, only one.

- We then develop algorithms that up to a dominant term match the lower bound.
- Key insight is that only up to two level of difficulty questions need to be asked, and in popular settings, only one.
- That is, after a quick exploration, the algorithm needs to settle at questions with close to a single level of difficulty

Single candidate - sequential, adaptive questions

$$P(success) = \frac{p}{p+x}$$

Prob. of success = p/(p+x)



Lower bound on all δ correct algorithms

- Through change of measure arguments that go back at least to Lai and Robbins 1985.

Lower bound on all δ correct algorithms

- Through change of measure arguments that go back at least to Lai and Robbins 1985.
- Under probability measure P, the ability of the candidate is $p \in [u_i, u_{i+1})$

Lower bound on all δ correct algorithms

- Through change of measure arguments that go back at least to Lai and Robbins 1985.
- Under probability measure P, the ability of the candidate is $p \in [u_i, u_{i+1})$
- Under probability measure $\tilde{P}$, the ability of the candidate is $u \notin [u_i, u_{i+1})$

Lower bound on all δ correct algorithms

- Through change of measure arguments that go back at least to Lai and Robbins 1985.
- Under probability measure P, the ability of the candidate is $p \in [u_i, u_{i+1})$
- Under probability measure $\tilde{P}$, the ability of the candidate is $u \notin [u_i, u_{i+1})$
- Question hardness x can be thought of as the arm pulled. Uncountably many

key inequality for developing lower bounds

- Adapting Kaufmann, Cappe, Garivier (2016) Lemma 1:

$$\sum_{x \in \mathcal{X}} E_P N_x \; KL\left(\frac{p}{p+x} \| \frac{u}{u+x}\right) \geq \log\left(\frac{1}{\delta}\right)$$

key inequality for developing lower bounds

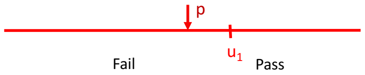
- Adapting Kaufmann, Cappe, Garivier (2016) Lemma 1:

$$\sum_{x \in \mathcal{X}} E_P N_x \; KL\left(\frac{p}{p+x} || \frac{u}{u+x}\right) \geq \log\left(\frac{1}{\delta}\right)$$

- We generalize to uncountably many questions

Single threshold setting linear program

Prob. of success = p/(p+x)



- Single threshold u_1, $p < u_1$, variables normalized by $\log\left(\frac{1}{\delta}\right)$

$$\min \quad \sum_x t_x$$

$$\text{s. t.} \quad \sum_x t_x \, KL\left(\frac{p}{p+x} \,\Big|\Big|\, \frac{u_1}{u_1+x}\right) \geq 1,$$

$$t_x \geq 0, \quad \forall x$$

- Linear program

$$\min \quad \sum_x t_x$$

$$\text{s. t.} \ \sum_x t_x KL\left(\frac{p}{p+x} || \frac{u_1}{u_1+x}\right) \geq 1,$$

$$t_x \geq 0, \quad \forall x.$$

- Solution

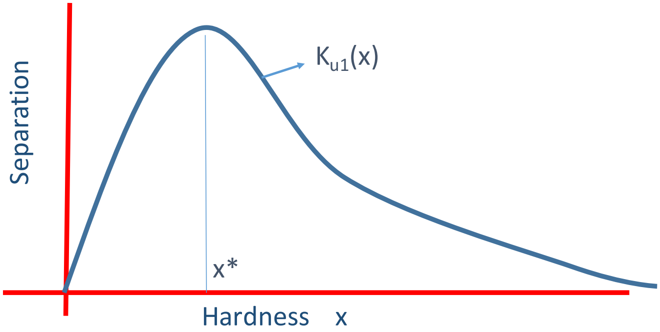
$$t_{x^*} = \frac{1}{KL\left(\frac{p}{p+x^*} || \frac{u_1}{u_1+x^*}\right)}$$

where

$$x^* = \arg\max_x KL\left(\frac{p}{p+x} || \frac{u_1}{u_1+x}\right).$$

Graphical view of maximum separation

$$x^* = \arg\max_x KL\left(\frac{p}{p+x} || \frac{u_1}{u_1+x}\right).$$



For multi-threshold, $p \in [u_1, u_2)$

$$
\begin{aligned}
\min \quad & \sum_x t_x \\
\text{s. t.} \quad & \sum_x t_x \, KL\left(\frac{p}{p+x} || \frac{u_1}{u_1+x}\right) \geq 1, \\
\text{and} \quad & \sum_x t_x \, KL\left(\frac{p}{p+x} || \frac{u_2}{u_2+x}\right) \geq 1
\end{aligned}
$$

For multi-threshold, $p \in [u_1, u_2)$

$$\min \quad \sum_x t_x$$

$$\text{s. t.} \quad \sum_x t_x \, KL\left(\frac{p}{p+x} || \frac{u_1}{u_1+x}\right) \geq 1,$$

$$\text{and} \quad \sum_x t_x \, KL\left(\frac{p}{p+x} || \frac{u_2}{u_2+x}\right) \geq 1$$

- Can restrict to atmost two t_x positive

For multi-threshold, $p \in [u_1, u_2)$

$$\min \quad \sum_x t_x$$

$$\text{s. t.} \ \sum_x t_x \, KL\left(\frac{p}{p+x} || \frac{u_1}{u_1+x}\right) \geq 1,$$

$$\text{and} \ \sum_x t_x \, KL\left(\frac{p}{p+x} || \frac{u_2}{u_2+x}\right) \geq 1$$

- Can restrict to atmost two t_x positive
- Re-expressing the constraints

$$(t_{x_1} + t_{x_2}) \min_{i=1,2} \sum_{j=1,2} \frac{t_{x_j}}{(t_{x_1} + t_{x_2})} KL\left(\frac{p}{p+x_j} || \frac{u_i}{u_i+x_j}\right) \geq 1$$

- Denoting
$$w = \frac{t_{x_1}}{t_{x_1} + t_{x_2}},$$
above simplifies to

$$\boxed{m^* \triangleq \max_{w \in [0,1], x_1, x_2} \min_{i=1,2} \left(w\, K_{u_i}(x_1) + (1-w) K_{u_i}(x_2)\right)}$$

where
$$K_{u_i}(x) \triangleq KL\left(\frac{p}{p+x} \middle|\middle| \frac{u_i}{u_i+x}\right)$$

- Denoting

$$w = \frac{t_{x_1}}{t_{x_1} + t_{x_2}},$$

above simplifies to

$$\boxed{m^* \triangleq \max_{w \in [0,1], x_1, x_2} \min_{i=1,2} \left(w\, K_{u_i}(x_1) + (1-w) K_{u_i}(x_2)\right)}$$

where

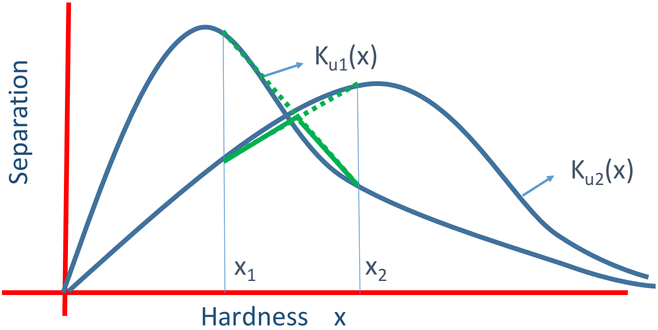
$$K_{u_i}(x) \triangleq KL\left(\frac{p}{p+x} \| \frac{u_i}{u_i+x}\right)$$

- Proposition

Sample complexity of any δ*−correct algorithm* $\geq \frac{1}{m^*} \log \frac{1}{\delta}$

Graphical view

$$m^* = \max_{w\in[0,1],x_1,x_2} \min_{i=1,2} \left(w\, K_{u_i}(x_1) + (1-w)K_{u_i}(x_2)\right)$$



The dual

- $$\max \; y_1 + y_2$$

  $$\text{s. t.} \quad y_1 K_{u_1}(x) + y_2 K_{u_2}(x) \leq 1, \text{for all } x$$

  $$y_1, y_2 \geq 0$$

The dual

- $$\max \; y_1 + y_2$$

  $$\text{s. t.} \quad y_1 K_{u_1}(x) + y_2 K_{u_2}(x) \leq 1, \text{for all } x$$

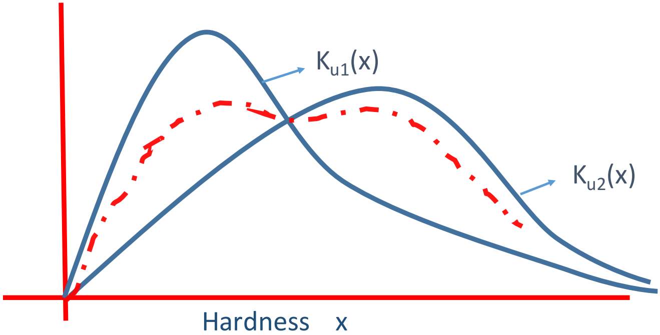
  $$y_1, y_2 \geq 0$$

- This simplifies to

  $$\boxed{m^* = \min_{\lambda \in [0,1]} \sup_x \left(\lambda K_{u_1}(x) + (1-\lambda) K_{u_2}(x)\right)}$$

Graphical view of the dual minimax problem

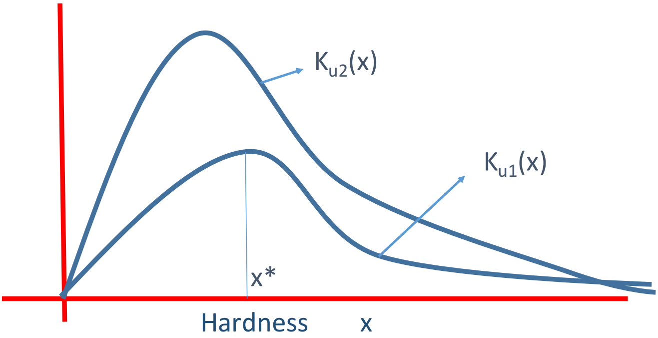
$$\min_{\lambda \in [0,1]} \sup_{x} \left(\lambda K_{u_1}(x) + (1-\lambda) K_{u_2}(x)\right)$$



Sufficient conditions for single question level optimality

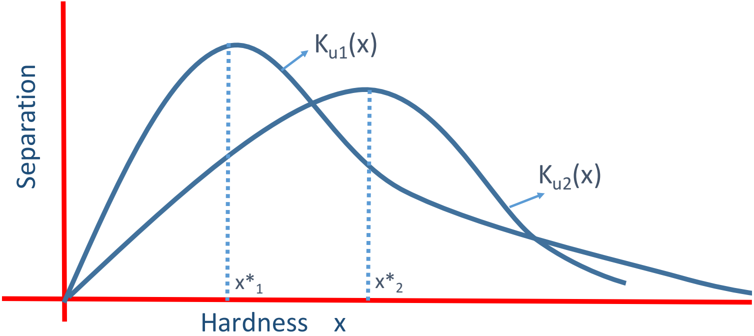
Dominant separation function

$$\min_{\lambda \in [0,1]} \sup_x \left(\lambda K_{u_1}(x) + (1-\lambda) K_{u_2}(x)\right)$$



Intersecting separating functions

$$\min_{\lambda \in [0,1]} \sup_{x} \left(\lambda K_{u_1}(x) + (1-\lambda) K_{u_2}(x)\right)$$



Intersecting separating functions

- Due to quasi-convexity of K_{u_1} and K_{u_2}, both the functions are increasing for $x < x_1^*$, and decreasing for $x > x_2^*$.

Intersecting separating functions

- Due to quasi-convexity of K_{u_1} and K_{u_2}, both the functions are increasing for $x < x_1^*$, and decreasing for $x > x_2^*$.
- Hence,

$$m^* = \inf_{\lambda \in [0,1]} \sup_{x \in [x_1^*, x_2^*]} \left(\lambda\, K_{u_1}(x) + (1-\lambda)\, K_{u_2}(x) \right).$$

Intersecting separating functions

- Due to quasi-convexity of K_{u_1} and K_{u_2}, both the functions are increasing for $x < x_1^*$, and decreasing for $x > x_2^*$.
- Hence,

$$m^* = \inf_{\lambda \in [0,1]} \sup_{x \in [x_1^*, x_2^*]} \left(\lambda K_{u_1}(x) + (1-\lambda) K_{u_2}(x) \right).$$

- Suppose that $K_{u_1}(x)$ is convex for $x \leq x_2^*$.

Intersecting separating functions

- Due to quasi-convexity of K_{u_1} and K_{u_2}, both the functions are increasing for $x < x_1^*$, and decreasing for $x > x_2^*$.
- Hence,

$$m^* = \inf_{\lambda \in [0,1]} \sup_{x \in [x_1^*, x_2^*]} \left(\lambda K_{u_1}(x) + (1-\lambda) K_{u_2}(x) \right).$$

- Suppose that $K_{u_1}(x)$ is convex for $x \leq x_2^*$.
- Then, $\lambda K_{u_1}(x) + (1-\lambda) K_{u_2}(x)$ is convex for $x \in [x_1^*, x_2^*]$.

Intersecting separating functions

- Due to quasi-convexity of K_{u_1} and K_{u_2}, both the functions are increasing for $x < x_1^*$, and decreasing for $x > x_2^*$.
- Hence,

$$m^* = \inf_{\lambda \in [0,1]} \sup_{x \in [x_1^*, x_2^*]} \left(\lambda\, K_{u_1}(x) + (1-\lambda)\, K_{u_2}(x) \right).$$

- Suppose that $K_{u_1}(x)$ is convex for $x \leq x_2^*$.
- Then, $\lambda\, K_{u_1}(x) + (1-\lambda)\, K_{u_2}(x)$ is convex for $x \in [x_1^*, x_2^*]$.
- By Sion's Minimax Theorem,

$$m^* = \sup_{x \in [x_1^*, x_2^*]} \min(K_{u_1}(x), K_{u_2}(x)).$$

Sufficient conditions for single question to be optimal

- **Result:** If the ratio $\frac{K'_{u_1}(x)}{K'_{u_2}(x)}$ is strictly decreasing in interval $[x_1^*, x_2^*]$ then the intersection point of the two curves $K_{u_1}(x)$ and $K_{u_2}(x)$ uniquely solves the dual problem.

Sufficient conditions for single question to be optimal

- **Result:** If the ratio $\frac{K'_{u_1}(x)}{K'_{u_2}(x)}$ is strictly decreasing in interval $[x_1^*, x_2^*]$ then the intersection point of the two curves $K_{u_1}(x)$ and $K_{u_2}(x)$ uniquely solves the dual problem.

- **Result:** If the response function h is of the form

$$h(p,x) = \frac{g(p)}{g(p) + k(x)}$$

for strictly increasing, positive functions g and k, then the ratio $\frac{K'_{u_1}(x)}{K'_{u_2}(x)}$ is strictly decreasing in interval $[x_1^*, x_2^*]$.

Sufficient conditions for single question to be optimal

- **Result:** If the ratio $\frac{K'_{u_1}(x)}{K'_{u_2}(x)}$ is strictly decreasing in interval $[x_1^*, x_2^*]$ then the intersection point of the two curves $K_{u_1}(x)$ and $K_{u_2}(x)$ uniquely solves the dual problem.

- **Result:** If the response function h is of the form

$$h(p, x) = \frac{g(p)}{g(p) + k(x)}$$

 for strictly increasing, positive functions g and k, then the ratio $\frac{K'_{u_1}(x)}{K'_{u_2}(x)}$ is strictly decreasing in interval $[x_1^*, x_2^*]$.

- Thus, single question is optimal for logit-type models.

An Asymptotically Optimal δ PAC-learning Algorithm

Sequential algorithm

- Adaptively asks a candidate questions $X_1, X_2, \ldots$ that are measurable relative to the filtration $\mathcal{F}_t$ generated by past questions $X_1, \ldots, X_{t-1}$ and responses $I_1, \ldots, I_{t-1}$

Sequential algorithm

- Adaptively asks a candidate questions $X_1, X_2, \ldots$ that are measurable relative to the filtration $\mathcal{F}_t$ generated by past questions $X_1, \ldots, X_{t-1}$ and responses $I_1, \ldots, I_{t-1}$
- At any stage $t = 1, 2, \ldots$, the algorithm also decides whether the stopping time $\tau = t$ or $\tau > t$

Sequential algorithm

- Adaptively asks a candidate questions $X_1, X_2, \ldots$ that are measurable relative to the filtration $\mathcal{F}_t$ generated by past questions $X_1, \ldots, X_{t-1}$ and responses $I_1, \ldots, I_{t-1}$
- At any stage $t = 1, 2, \ldots$, the algorithm also decides whether the stopping time $\tau = t$ or $\tau > t$
- If the algorithm decides to continue, then it must also determine X_{t+1}, the level of difficulty of the next question.

Sequential algorithm

- Adaptively asks a candidate questions $X_1, X_2, \ldots$ that are measurable relative to the filtration $\mathcal{F}_t$ generated by past questions $X_1, \ldots, X_{t-1}$ and responses $I_1, \ldots, I_{t-1}$

- At any stage $t = 1, 2, \ldots$, the algorithm also decides whether the stopping time $\tau = t$ or $\tau > t$

- If the algorithm decides to continue, then it must also determine X_{t+1}, the level of difficulty of the next question.

- If the former, it announces that the candidate's ability lies in the interval $[u_J, u_{J+1})$ for some J.

First identifying MLE

- Likelihood of observing data $(I_j : 1 \leq i \leq t)$ when the underlying ability is p and the questions are asked at level $\mathbf{X}_t$

$$L(p; \mathbf{X}_t) = \prod_{j=1}^{t} \left(\frac{p}{p + X_j} \right)^{I_j} \left(\frac{X_j}{p + X_j} \right)^{1 - I_j}$$

First identifying MLE

- Likelihood of observing data $(I_j : 1 \leq i \leq t)$ when the underlying ability is p and the questions are asked at level $\mathbf{X}_t$

$$L(p; \mathbf{X}_t) = \prod_{j=1}^{t} \left(\frac{p}{p + X_j}\right)^{I_j} \left(\frac{X_j}{p + X_j}\right)^{1 - I_j}$$

- The log-likelihood equals

$$\sum_{j=1}^{t} I_j \log\left(\frac{p}{p + X_j}\right) + (1 - I_j) \log\left(\frac{X_j}{p + X_j}\right).$$

First identifying MLE

- Likelihood of observing data $(I_j : 1 \leq i \leq t)$ when the underlying ability is p and the questions are asked at level $\mathbf{X}_t$

$$L(p; \mathbf{X}_t) = \prod_{j=1}^{t} \left(\frac{p}{p + X_j} \right)^{I_j} \left(\frac{X_j}{p + X_j} \right)^{1 - I_j}$$

- The log-likelihood equals

$$\sum_{j=1}^{t} I_j \log \left(\frac{p}{p + X_j} \right) + (1 - I_j) \log \left(\frac{X_j}{p + X_j} \right).$$

- Thus, the maximum likelihood estimator (mle) $\hat{p}_t$ uniquely solves

$$\sum_{j=1}^{t} \frac{p}{p + X_j} = \sum_{j=1}^{t} I_j.$$

Stopping rule based on generalized likelihood ratio test

- Consider the ratio of the likelihood of observing the data under MLE with the likelihood under the most likely alternative hypothesis. The algorithm stops when this ratio is sufficiently large

Stopping rule based on generalized likelihood ratio test

- Consider the ratio of the likelihood of observing the data under MLE with the likelihood under the most likely alternative hypothesis. The algorithm stops when this ratio is sufficiently large
- If MLE is $\hat{p}_t \in (u_i, u_{i+1})$, its likelihood equals $L(\hat{p}_t; \mathbf{X}_t)$

Stopping rule based on generalized likelihood ratio test

- Consider the ratio of the likelihood of observing the data under MLE with the likelihood under the most likely alternative hypothesis. The algorithm stops when this ratio is sufficiently large
- If MLE is $\hat{p}_t \in (u_i, u_{i+1})$, its likelihood equals $L(\hat{p}_t; \mathbf{X}_t)$
- The likelihood of the most likely alternative hypothesis corresponds to $\max(L(u_i, \mathbf{X}_t), L(u_{i+1}, \mathbf{X}_t))$

Stopping rule based on generalized likelihood ratio test

- Consider the ratio of the likelihood of observing the data under MLE with the likelihood under the most likely alternative hypothesis. The algorithm stops when this ratio is sufficiently large
- If MLE is $\hat{p}_t \in (u_i, u_{i+1})$, its likelihood equals $L(\hat{p}_t; \mathbf{X}_t)$
- The likelihood of the most likely alternative hypothesis corresponds to $\max(L(u_i, \mathbf{X}_t), L(u_{i+1}, \mathbf{X}_t))$
- The stopping rule corresponds to the log-likelihood ratio, that is,

$$\min_{u \in \{u_i, u_{i+1}\}} \left[\sum_{j=1}^{t} I_j \log\left(\frac{\hat{p}_t/(\hat{p}_t + X_j)}{u/(u+X_j)} \right) + (1 - I_j) \log\left(\frac{u + X_j}{\hat{p}_t + X_j} \right) \right]$$

exceeding a threshold $\beta(t, \delta) = \log(\frac{ct^\alpha}{\delta})$

The question selection rule

- Suppose the algorithm has proceeded for t steps, with $\mathbf{X}_t = (X_1, \ldots, X_t)$ denoting the level of difficulty of questions asked

The question selection rule

- Suppose the algorithm has proceeded for t steps, with $\mathbf{X}_t = (X_1, \ldots, X_t)$ denoting the level of difficulty of questions asked

- Next question determined by solving the lower bound optimization problem, with $\hat{p}_t$ in place of p, and finding questions levels $x_1(\hat{p}_t)$ and $x_2(\hat{p}_t)$ with weights $w(\hat{p}_t)$ and $1 - w(\hat{p}_t)$

The question selection rule

- Suppose the algorithm has proceeded for t steps, with $\mathbf{X}_t = (X_1, \ldots, X_t)$ denoting the level of difficulty of questions asked
- Next question determined by solving the lower bound optimization problem, with $\hat{p}_t$ in place of p, and finding questions levels $x_1(\hat{p}_t)$ and $x_2(\hat{p}_t)$ with weights $w(\hat{p}_t)$ and $1 - w(\hat{p}_t)$
- Next question level X_{t+1} is set equal to $x_1(\hat{p}_t)$ with probability $w(\hat{p}_t)$ and to $x_2(\hat{p}_t)$ otherwise.

The question selection rule

- Suppose the algorithm has proceeded for t steps, with $\mathbf{X}_t = (X_1, \ldots, X_t)$ denoting the level of difficulty of questions asked

- Next question determined by solving the lower bound optimization problem, with $\hat{p}_t$ in place of p, and finding questions levels $x_1(\hat{p}_t)$ and $x_2(\hat{p}_t)$ with weights $w(\hat{p}_t)$ and $1 - w(\hat{p}_t)$

- Next question level X_{t+1} is set equal to $x_1(\hat{p}_t)$ with probability $w(\hat{p}_t)$ and to $x_2(\hat{p}_t)$ otherwise.

- After observing I_{t+1} one again checks whether the stopping rule holds or whether the algorithm continues.

Formal result

- Proposition

  Let $\tau(\delta)$ denote the stopping time and $p \in [u_i, u_{i+1}]$. Then the following two properties are satisfied:

  a) **Sample complexity**

  $$\lim_{\delta \to 0} \frac{E_P[\tau(\delta)]}{\log \delta} = -m^*.$$

  b) **δ-PAC Property**

  $$P\left(\hat{p}_\tau \notin [u_i, u_{i+1}]\right) \leq \delta.$$

Conclusions

- We reviewed the evolving literature on regret minimization

Conclusions

- We reviewed the evolving literature on regret minimization
- We analyzed the partition identification problem

Conclusions

- We reviewed the evolving literature on regret minimization
- We analyzed the partition identification problem
- We discussed the interview problem.