Week 6:
Multi-armed Bandit

AIM-5014-1A: Experimental Optimization

David Sweet // 20230706

Review: LLN, CLT, A/B Testing

- As $N \to \infty$
  - LLN: $\bar{y} \to E[y_i]$, measured BM approaches “true” BM
  - CLT: $\bar{y} \sim \mathcal{N}(E[y_i], \sigma^2)$, measured BM is normal (gaussian)
- **Design**: $N \geq \left(\frac{2.5\hat{\sigma}_\delta}{PS}\right)^2$
- **Measure**: Randomize, $\bar{\delta} = \bar{y}_B - \bar{y}_A, \quad se = \sigma_\delta / \sqrt{N}$
- **Analyze**: If $\bar{\delta} > PS$ and $\frac{\bar{\delta}}{se} \geq 1.64$ (and guardrails ok), then accept B.

Review: False Positive Traps

- **Don’t stop early**, even if t-stat looks good
- **Beware multiple comparisons** in A/B/C/… tests
  - Use Bonferroni correction: p = 0.05 / (K-1)
  - Then accept if: $\bar{\delta} > PS$ and $t = \frac{\bar{\delta}}{se} \geq 1.64$

Case: Two HFT strategies

- Two strategies, A and B; 100 stocks
- PS = $100 / day
- $\hat{\sigma}_\delta = \$125/\sqrt{\text{day}}$

No, really, “dollars per square-root day”

- $N = \left(\frac{2.5\hat{\sigma}_\delta}{PS}\right)^2 = \left(\frac{2.5 \times \$125/\sqrt{\text{day}}}{\$100/\text{day}}\right)^2$ = 10 days
- Allocate 50 stocks to A, 50 stocks to B
- Run A/B test for two weeks.

Case: Two HFT strategies

- At end, hindsight regret:
  - “If we’d run A the whole time we’d have made more money.”, or
  - “If we’d run B the whole time we’d have made more money.”
- Could you choose winner earlier?
  - Don’t stop early; **early stopping** increases false positives
- Solution: “ease into it”

Monitor & Adapt

- Each day: Calculate $\bar{y}_a, se_a, \bar{y}_b, se_b$ so far
- From $\bar{y}_a, se_a, \bar{y}_b, se_b$estimate probability A is better than B
  - $p_a = P\{\text{A better}\} = P\{E[y_a] > E[y_b]\}$
  - $p_b = 1 - p_a$
- Allocate $50 \times p_a$ stocks to A, $50 \times p_b$ stocks to B
- Start at 50/50 and gradually transition towards better strategy

Monitor & adapt

- Start at 50/50 and gradually transition towards better strategy
  - Spend more time trading better strategy
  - ==> Higher pnl during experiment
  - **Reduces experimentation cost**
- All experimental methods invented to reduce experimentation cost

Probability A is better

- Recall: $y_a \sim \mathcal{N}(E[y_a], \sigma_a^2)$ and $y_b \sim \mathcal{N}(E[y_b], \sigma_b^2)$
- Approximate normal dists by $\mathcal{N}(\bar{y}_a, se_a)$ and $\mathcal{N}(\bar{y}_b, se_b)$
- Draw 10,000 samples from each of $a \sim \mathcal{N}(\bar{y}_a, se_a)$ and $b \sim \mathcal{N}(\bar{y}_b, se_b)$
- Estimate: $p_a = \dfrac{\text{\# times } a > b}{10{,}000}$ and $p_b = \dfrac{\text{\# times } b > a}{10{,}000} = 1 - p_a$

Probability A is better

- Draw 10,000 samples from each of $a \sim \mathcal{N}(\bar{y}_a, se_a)$ and $b \sim \mathcal{N}(\bar{y}_b, se_b)$
- Estimate: $p_a = \frac{\text{\# times } a > b}{10{,}000}$ and $p_b = \frac{\text{\# times } b > a}{10{,}000} = 1 - p_a$

```
a = y_a + se_a*np.random.normal(size=(10000,))
b = y_b + se_b*np.random.normal(size=(10000,))

p_a = (a>b).mean()
p_b = (b>a).mean()
```

Probability an arm is best

- Works for multiple arms, too; A/B/C/…
- $a_k \sim \mathcal{N}(\bar{y}_k, se_k); p_{a_k} = \dfrac{\text{\# times } a_k > a_{\sim k}}{10{,}000}$

```
a = y_a + se_a*np.random.normal(size=(10000,))
b = y_b + se_b*np.random.normal(size=(10000,))
c = y_c + se_c*np.random.normal(size=(10000,))

p_a = ( (a>b) & (a>c) ).mean()
p_b = ( (b>a) & (b>c) ).mean()
p_c = ( (c>a) & (c>b) ).mean()
```

One weird trick

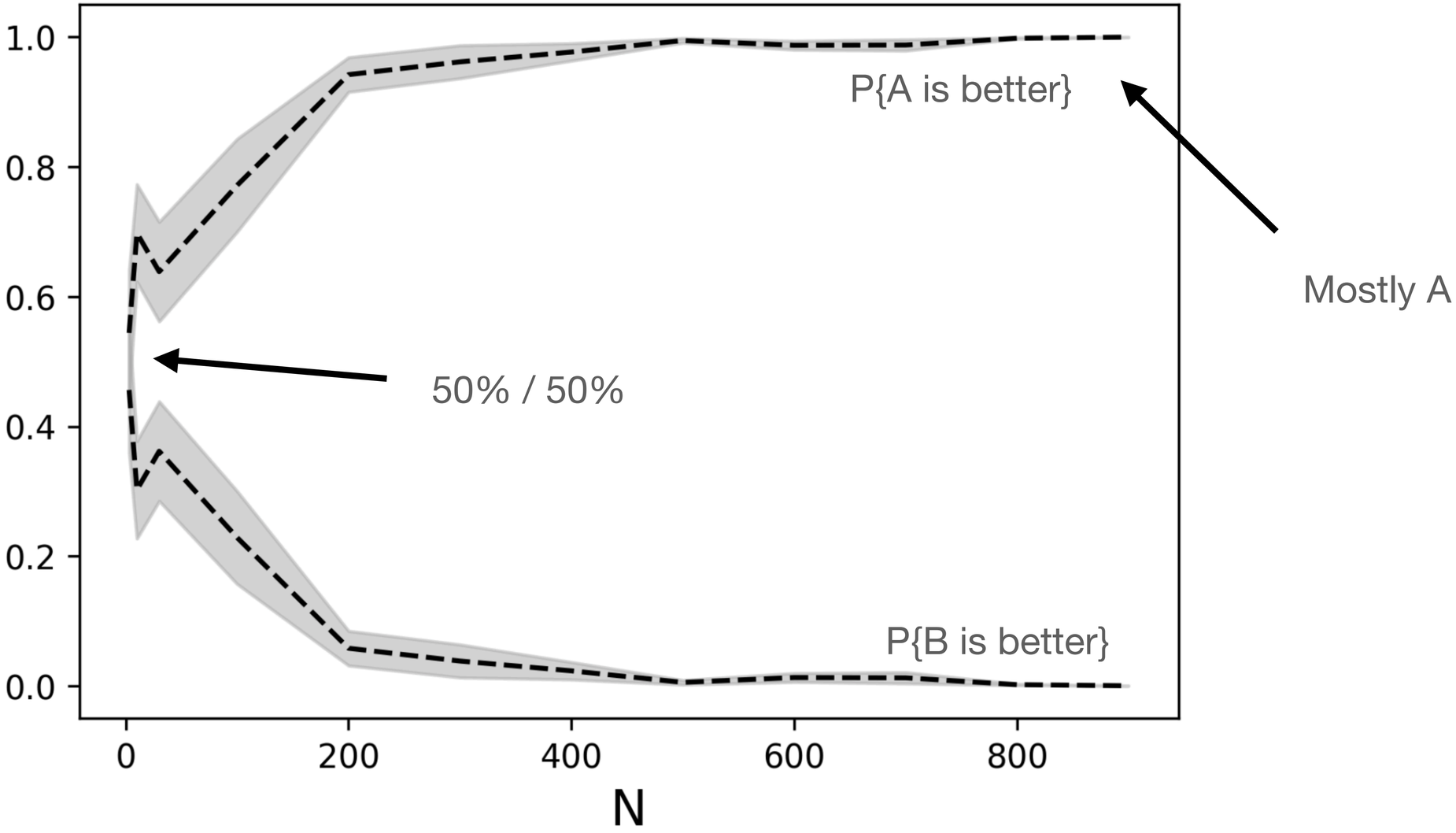
- Draw **one** sample from each of $a \sim \mathcal{N}(\bar{y}_a, se_a)$ and $b \sim \mathcal{N}(\bar{y}_b, se_b)$
- P{a>b} = p_a, by definition
- Assign each stock by a single pair of draws
- Noisier, but faster:
  1 draw instead of 10,000
- Called *Thompson Sampling*

```
for i_stock in (100):
    a = y_a + se_a*np.random.normal(size=(1,))
    b = y_b + se_b*np.random.normal(size=(1,))
    if a > b:
        # assign stock i_stock to A
    else:
        # assign stock i_stock to B
```

Experiment on an app

- Two ML recommender models, A and B; BM = time spent
- User opens your app, should you show A or B?
- Instead of randomizing 50/50, try Thompson Sampling
  - Draw **one** sample from each of $a \sim \mathcal{N}(\bar{y}_a, se_a)$ and $b \sim \mathcal{N}(\bar{y}_b, se_b)$
  - If $a > b$, show A, else show B
  - Use time spent to update stats: $\bar{y}_a, se_a, \bar{y}_b, se_b$
- Sequential decision-making; adapting / progressively improving

Experiment with Thompson sampling



Thompson sampling with three arms

http://andamooka.org/~dsweet/EOCourse/Thompson_Sampling.mov

Thompson Sampling: Stopping

- Stop when $\max\{p_k\} > 0.95$
  - When the largest “P{best arm}” is very large
- Just run the best arm
  - 5% you’re wrong

Recap: Thompson Sampling

- Method:
  - **Draw once** from each arm's model of BM: $\{\mathcal{N}(\bar{y}_k, se_k)\}$
  - Run arm $k^* = \arg\max\{\mathcal{N}(\bar{y}_k, se_k)\}$
  - Arms are run $\propto$ probability of being best
- Advantages:
  - Fast b/c only one draw from each arm's model
  - Increases BM **while experimenting**, reduces experimentation cost

One-Armed Bandit

No arm here

Only one arm

“Bandit” == Steals your money

This is a slot machine

Multi-Armed Bandit Problem

- “Multi-Armed” Bandit ==> Imagine a slot machine with multiple arms?
  - Doesn’t really make sense.
- Instead, imagine multiple slot machines
  - Each has one arm that you can pull
  - The distribution of the payout differs from machine to machine
  - You pull one arm at a time
  - Your goal is to earn as much (or lose as little) money as possible

Multi-Armed Bandit Problem

- K arms, $\{a_k\}$
- Payout, aka *reward*, is some distribution, $f(a_k)$
- Find sequence of arms, $a_{k_1}, a_{k_2}, a_{k_3}, \ldots, a_{k_N}$
  - that maximizes $\sum_{i}^{N} f(a_{k_i})$
- You only get to try one sequence, so choose each a_{k_i} wisely

Multi-Armed Bandit (MAB) Problem

- K arms, $\{a_k\}$ == different versions, A/B/C/…
- Thompson Sampling is one (great) solution to the MAB problem
  - Pull arm a_k that has highest $p_k = P\{a_k \text{ is best}\}$
    - i.e., assign user to arm A/B/C/…
  - Receive reward, $f(a_k)$
    - i.e., record “time spent by user”

This is why we call them “arms”

Multi-Armed Bandit (MAB) Problem

MAB	Experimentation
$\{a_k\}$	A/B/C/…; treatments
$a_{k_1}, a_{k_2}, a_{k_3}, \ldots, a_{k_N}$	Interventions
$f(a_k)$	Business Metric

Multi-Armed Bandit (MAB) Problem

- Epsilon-greedy, randomize:
  - 90% run arm w/best $\bar{y}_k$ so far
  - 10% run a randomly-chosen arm
- UCB:
  - Run arm with largest $\bar{y}_k + se$
  - “Optimism under uncertainty”

Multi-Armed Bandit Problem

“Originally considered by Allied scientists in World War II, it proved so intractable that, according to Peter Whittle, the problem was proposed to be dropped over Germany so that German scientists could also waste their time on it.”

https://en.wikipedia.org/wiki/Multi-armed_bandit

On the Likelihood that One Unknown Probability Exceeds Another
in View of the Evidence of Two Samples
William Thompson, 1933

https://www.jstor.org/stable/2332286

History of Thompson Sampling

- 1933: Thompson sampling introduced
  https://www.jstor.org/stable/2332286
- 1940's: "MAB is impossibly hard!"
  https://en.wikipedia.org/wiki/Multi-armed_bandit
- 2002: UCB is asymptotically optimal
  https://link.springer.com/article/10.1023/a:1013689704352
- 2012-2017: Thompson sampling is optimal, and better than UCB in practice
  http://proceedings.mlr.press/v23/agrawal12/agrawal12.pdf
  https://arxiv.org/abs/1209.3353
  http://www.columbia.edu/~sa3305/papers/j3-corrected.pdf

also 2002: eps-greedy can be hacked
to be asymptotically optimal,
but it's clumsy to use

Summary

- MAB methods maximize BM during experiment
  - by adapting to measurements
- Thompson Sampling is simple, optimal, and empirically performant
  - **Draw once** from each arm's model of BM: $\{\mathcal{N}(\bar{y}_k, se_k)\}$
  - Run arm $k^* = \arg\max\{\mathcal{N}(\bar{y}_k, se_k)\}$
  - Arms are run $\propto$ probability of being best