Artwork Personalization at Netflix

Justin Basilico
QCon SF 2018
2018-11-05

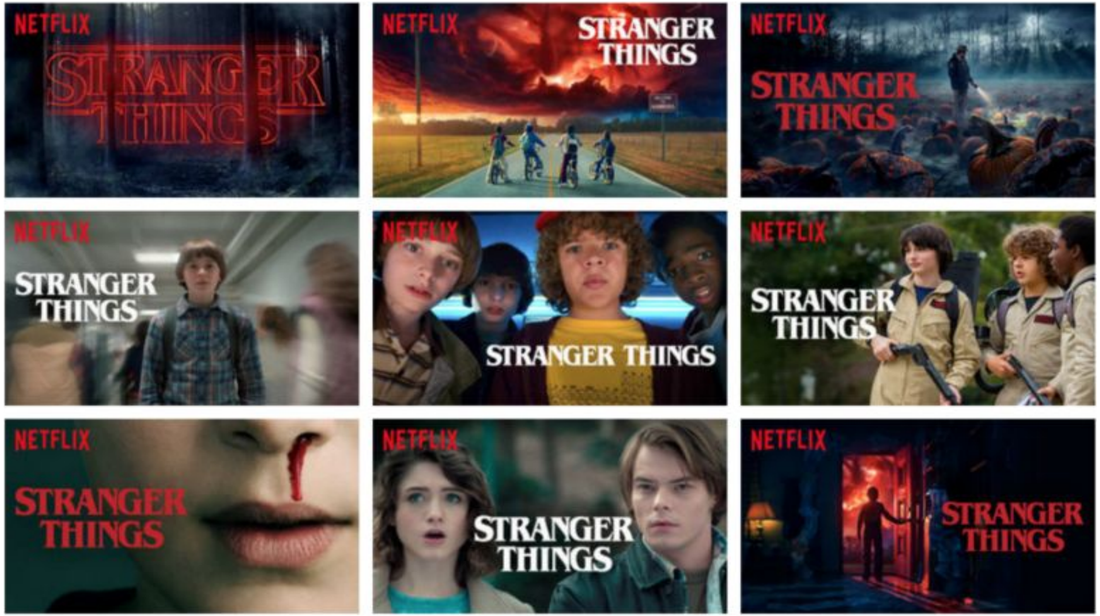
@JustinBasilico

NETFLIX

QCon SAN FRANCISCO by InfoQ


NETFLIX ORIGINAL
STRANGER THINGS
95% Match 2016 1 Season 4K Ultra HD 5.1
When a young boy vanishes, a small town uncovers a mystery involving secret experiments, terrifying supernatural forces and one strange little girl.
Winona Ryder, David Harbour, Matthew Modine
TV Shows, TV Sci-Fi & Fantasy, Teen TV Shows
STRANGER THINGS
Popular on Netflix
STRANGER THINGS
BRIGHT
MARVEL'S THE PUNISHER
MINDHUNTER
THE CROWN
Recently Watched
THE MEYEROWITZ STORIES (NEW AND SELECTED)
AMERICAN VANDAL
STAR TREK: DISCOVERY
FULLER HOUSE
NETFLIX

Which artwork to show?



A **good image** is...

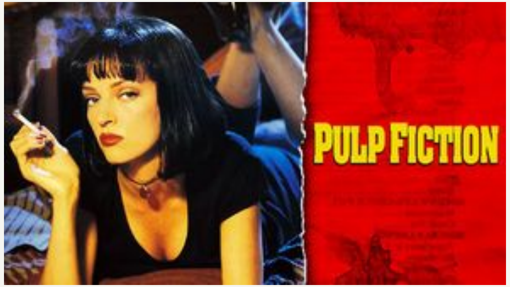
1. Representative
2. Informative
3. Engaging
4. Differential

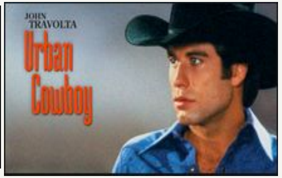
A **good image** is...

1. Representative
2. Informative
3. Engaging
4. Differential

Personal

Intuition: Preferences in cast members

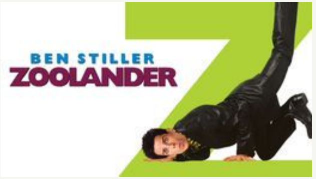
→


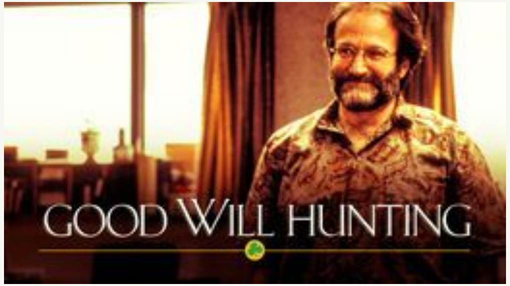


→


NETFLIX

Intuition: Preferences in genre



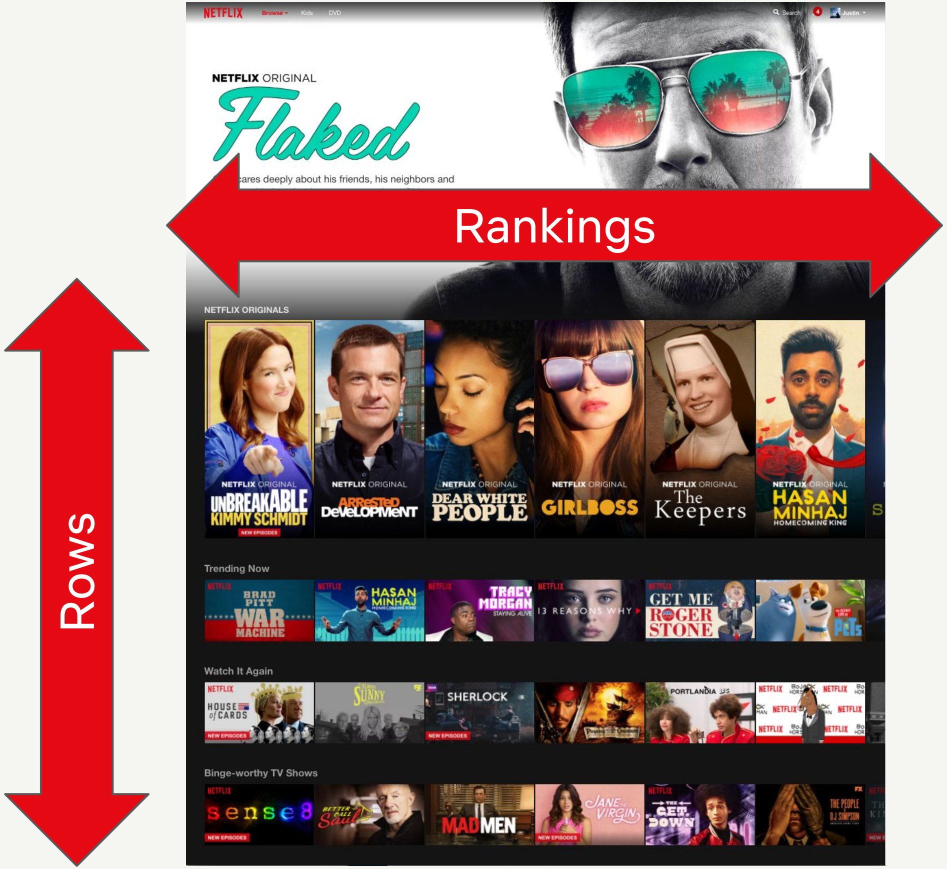


Choose artwork so that members understand if they will likely enjoy a title to maximize satisfaction and retention

NETFLIX

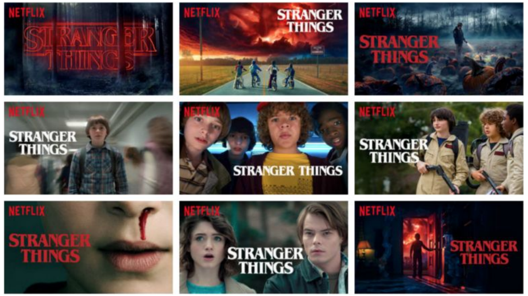
Challenges in Artwork Personalization

Everything is a Recommendation



Over 80% of what people watch comes from our recommendations

Attribution



Pick
only one

Was it the recommendation or artwork?
Or both?

Change Effects

Day 1

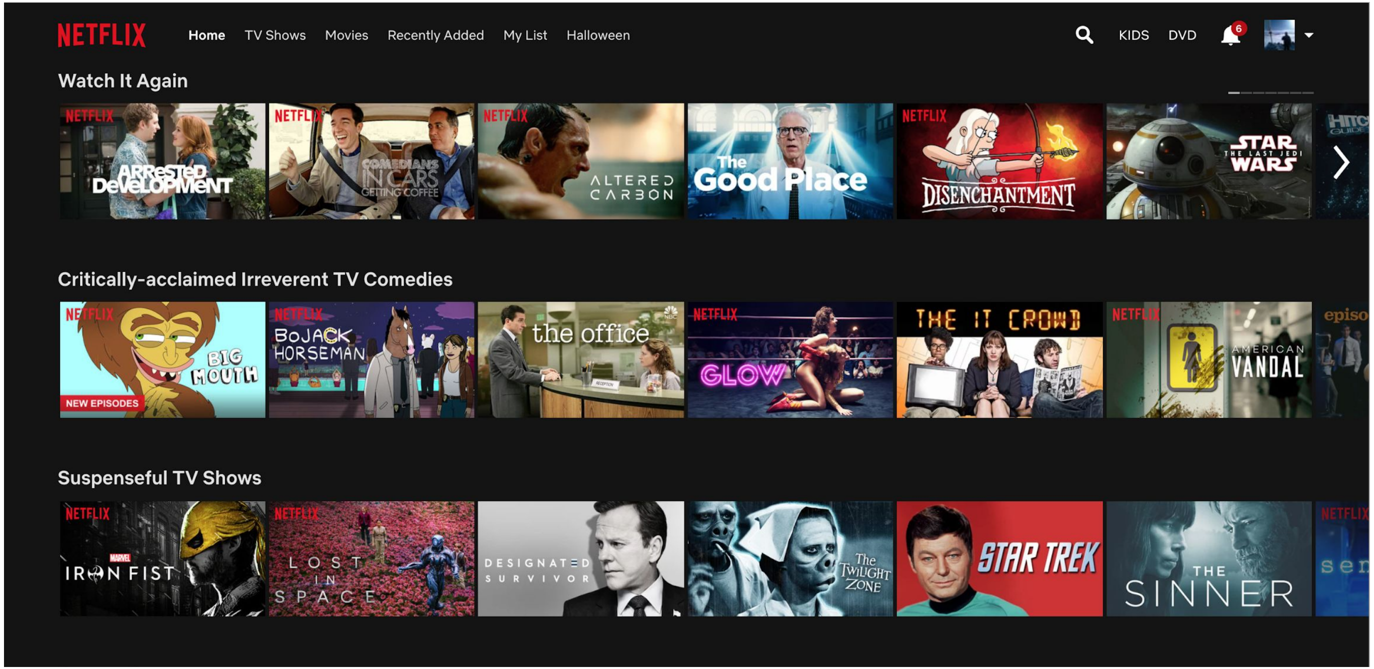
Day 2

Which one caused the play?
Is change confusing?

Adding meaning and avoiding clickbait

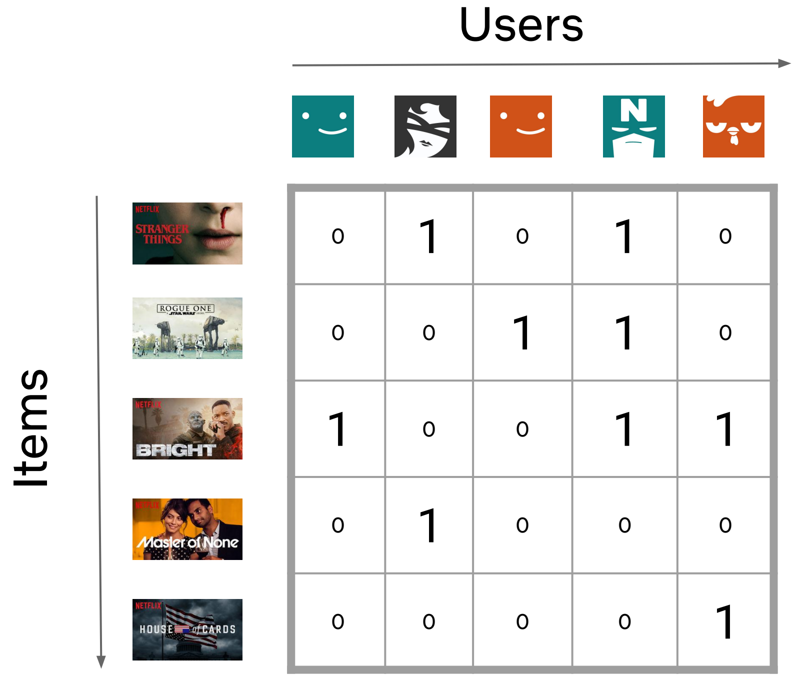
- Creatives select the images that are available
- But algorithms must be still robust

Scale



Over 20M RPS for images at peak

Traditional Recommendations



0	1	0	1	0
0	0	1	1	0
1	0	0	1	1
0	1	0	0	0
0	0	0	0	1

Collaborative Filtering:

Recommend items that similar users have chosen

Members can only play images we choose

Need something more

NETFLIX

Bandit


Not that kind of Bandit
Smokey AND THE Bandit

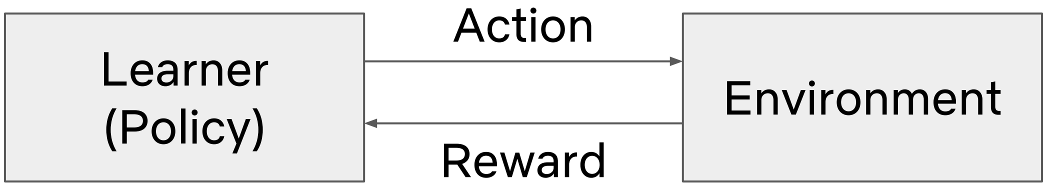
Image from [Wikimedia commons]

Multi-Armed Bandits (MAB)

- Multiple slot machines with **unknown reward distribution**
- A gambler can play one arm at a time
- Which machine to play to maximize reward?

NETFLIX

Bandit Algorithms Setting



Each round:

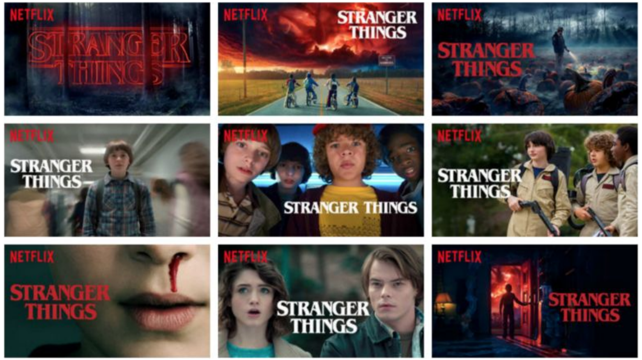
- Learner chooses an **action**
- Environment provides a real-valued **reward** for action
- Learner updates to **maximize the cumulative reward**

Artwork Optimization as Bandit

- **Environment**: Netflix homepage
- **Learner**: Artwork selector for a show
- **Action**: Display specific image for show
- **Reward**: Member has positive engagement

Images as Actions

- What images should creatives provide?
    - Variety of image designs
    - Thematic and visual differences
- How many images?
    - Creating each image has a cost
    - Diminishing returns

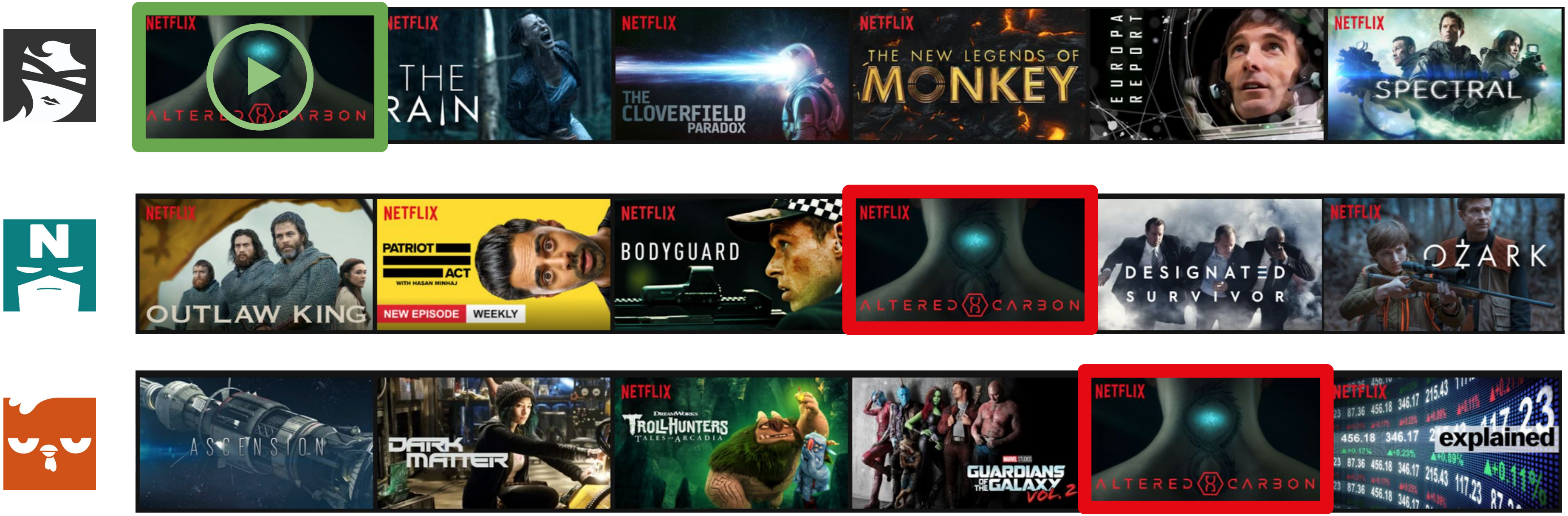


Designing Rewards

- What is a **good outcome**?
  - ✓ Watching and enjoying the content
- What is a **bad outcome**?
  - ✖ No engagement
  - ✖ Abandoning or not enjoying the content

Metric: Take Fraction

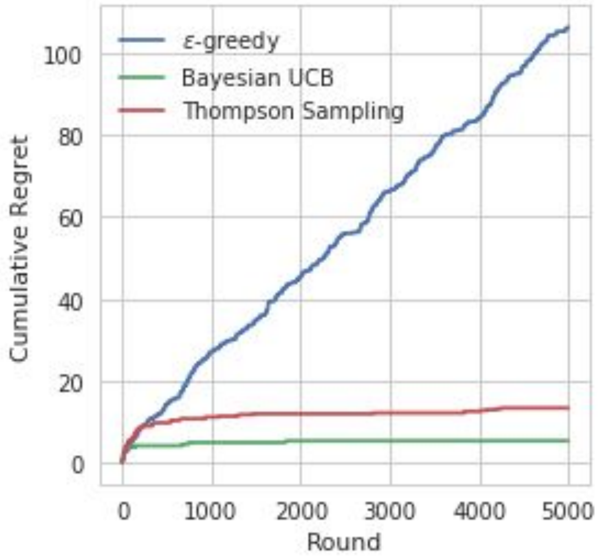
Example: Altered Carbon



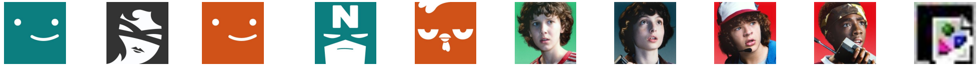
Take Fraction: 1/3

Minimizing Regret

- What is the best that a bandit can do?
    - Always choose optimal action
- **Regret**: Difference between optimal action and chosen action
- To maximize reward, **minimize the cumulative regret**



Bandit Example

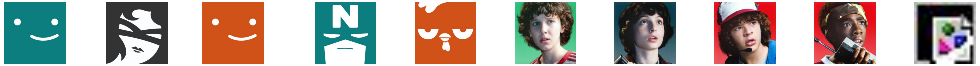


1			0			1		0	?
	0				0				?
		0		1			0		?

Actions

Historical rewards

Bandit Example



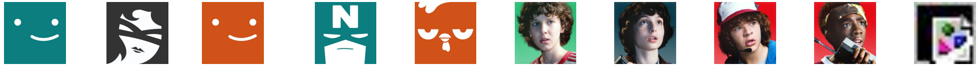
1			0			1		0	?
	0				0				?
		0		1			0		?

Choose image

Actions

Historical rewards

Bandit Example



Actions											Observed Take Fraction
Action 1	1			0			1		0	?	2/4
Action 2		0				0				?	0/2
Action 3			0		1			0		?	1/3

Overall: 3/9

Actions

Historical rewards

NETFLIX

Strategy

Show current best image vs. Try another image to learn if it is actually better

Maximization **Exploration**

Principles of Exploration

- Gather information to make the best overall decision in the long-run
- Best long-term strategy **may involve short-term sacrifices**

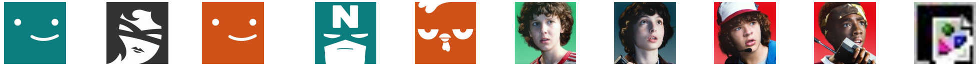
Common strategies

1. **Naive Exploration**
2. **Optimism in the Face of Uncertainty**
3. **Probability Matching**

Naive Exploration: ϵ-greedy

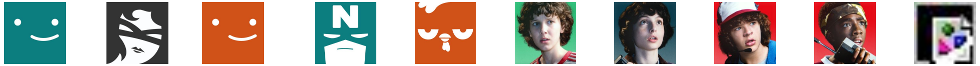
- **Idea: Add a noise to the greedy policy**
- Algorithm:
  - With probability ϵ
    - Choose one action uniformly at random
  - Otherwise
    - Choose the action with the best reward so far
- Pros: Simple
- Cons: Regret is unbounded

Epsilon-Greedy Example



											Observed Reward
	1			0			1		0	?	2/4 (greedy)
		0				0				?	0/2
			0		1			0		?	1/3

Epsilon-Greedy Example



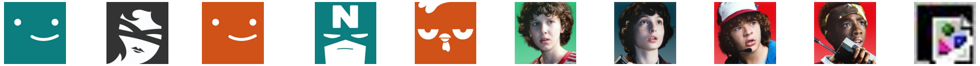
1			0			1		0	?
	0				0				?
		0		1			0		?

$1 - 2\epsilon / 3$

$\epsilon / 3$

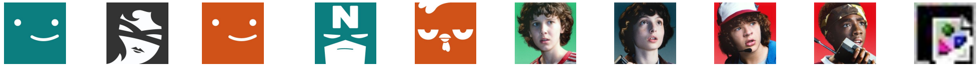
$\epsilon / 3$

Epsilon-Greedy Example



	1			0			1		0	?
		0				0				?
			0		1			0		?

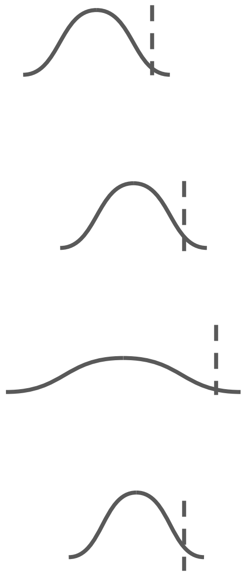
Epsilon-Greedy Example



											Observed Reward
	1			0			1		0		2/4 (greedy)
		0				0				0	**0/3**
			0		1			0			1/3

Optimism: Upper Confidence Bound (UCB)

- **Idea: Prefer actions with uncertain values**
- Approach:
  - Compute confidence interval of observed rewards for each action
  - Choose action **a** with the highest α-percentile
  - Observe reward and update confidence interval for **a**
- Pros: Theoretical regret minimization properties
- Cons: Needs to update quickly from observed rewards



Beta-Bernoulli Distribution

Beta

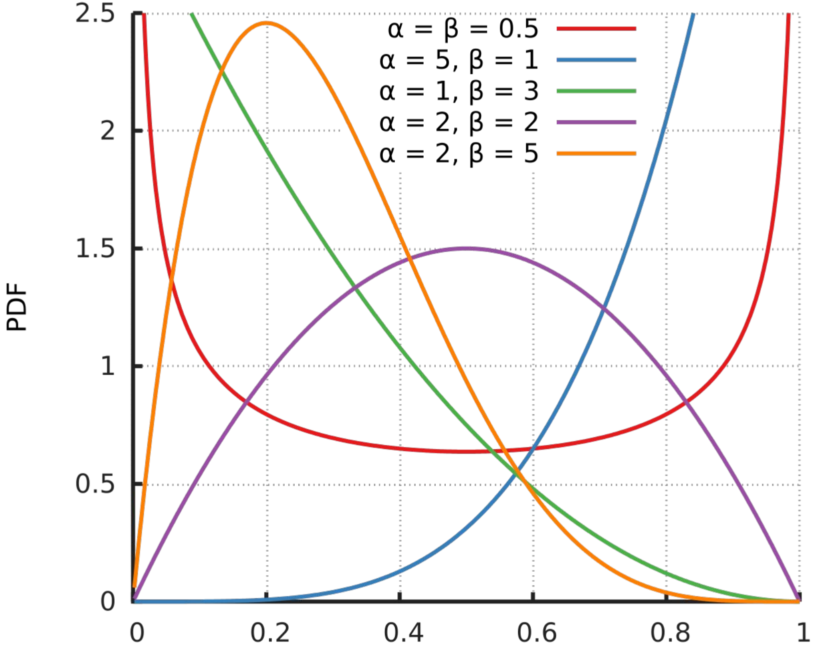


Image from [Wikipedia]

Bernoulli

Prior

Pr(1) = p

Pr(0) = 1 - p

Bandit Example with Beta-Bernoulli

Observed Take Fraction

Prior: $\beta(1, 1)$ +

A 

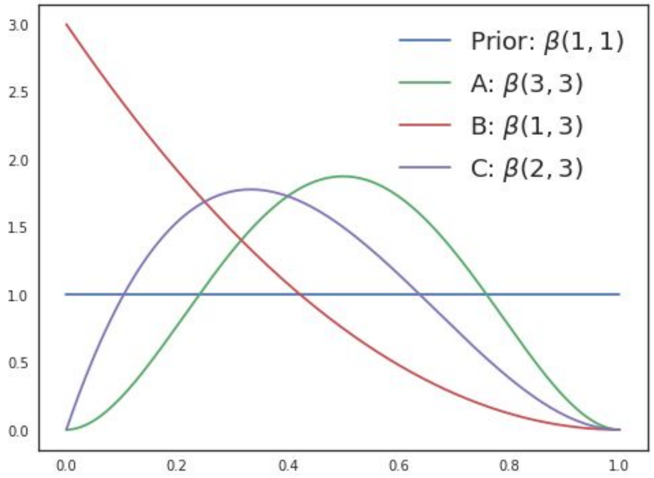
2/4 $\beta(3, 3)$

B 

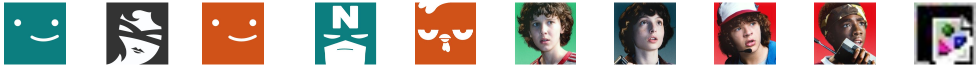
0/2 = $\beta(1, 3)$

C 

1/3 $\beta(2, 3)$

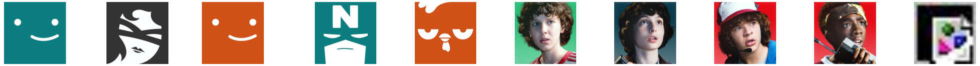


Bayesian UCB Example



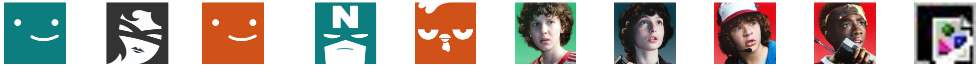
										Reward 95% Confidence
1			0			1		1	?	[0.15, 0.85]
	0				0				?	[0.01, 0.71]
		0		1			0		?	[0.07, 0.81]

Bayesian UCB Example



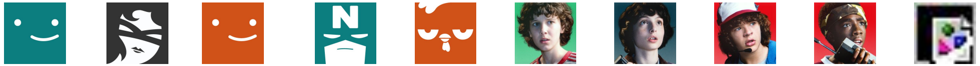
											Reward 95% Confidence
	1			0			1		1	?	[0.15, **0.85**]
		0				0				?	[0.01, 0.71]
			0		1			0		?	[0.07, 0.81]

Bayesian UCB Example



											Reward 95% Confidence
	1			0			1		1	0	**[0.12, 0.78]** 
		0				0					[0.01, 0.71] 
			0		1			0			[0.07, 0.81] 

Bayesian UCB Example



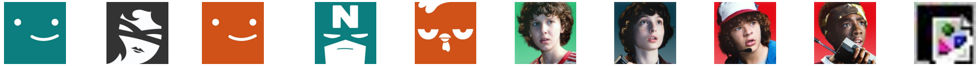
											Reward 95% Confidence
	1			0			1		1	0	[0.12, 0.78]
		0				0					[0.01, 0.71]
			0		1			0			[0.07, **0.81**]

NETFLIX

Probabilistic: Thompson Sampling

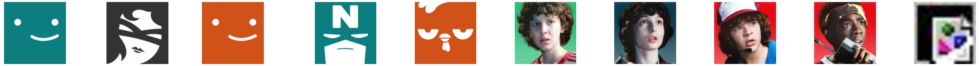
- **Idea: Select the actions by the probability they are the best**
- Approach:
  - Keep a distribution over model parameters for each action
  - Sample estimated reward value for each action
  - Choose action **a** with maximum sampled value
  - Observe reward for action **a** and update its parameter distribution
- Pros: Randomness continues to explore without update
- Cons: Hard to compute probabilities of actions

Thompson Sampling Example



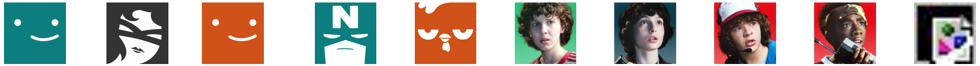
											Distribution
	1			0			1		0	?	$\beta(3, 3)$ = 
		0				0				?	$\beta(1, 3)$ = 
			0		1			0		?	$\beta(2, 3)$ = 

Thompson Sampling Example



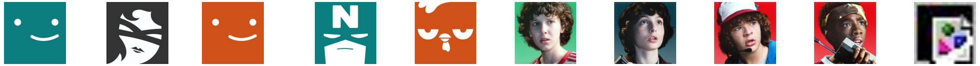
											Sampled values
	1			0			1		0	?	0.38
		0				0				?	0.18
			0		1			0		?	0.59

Thompson Sampling Example



											Sampled values
	1			0			1		0	?	0.38
		0				0				?	0.18
			0		1			0		?	**0.59**

Thompson Sampling Example



											Distribution	
	1			0			1		0		$\beta(3, 3)$ =	
		0				0					$\beta(1, 3)$ =	
			0		1			0		1	$\boldsymbol{\beta(3, 3)}$ =	

Many Variants of Bandits

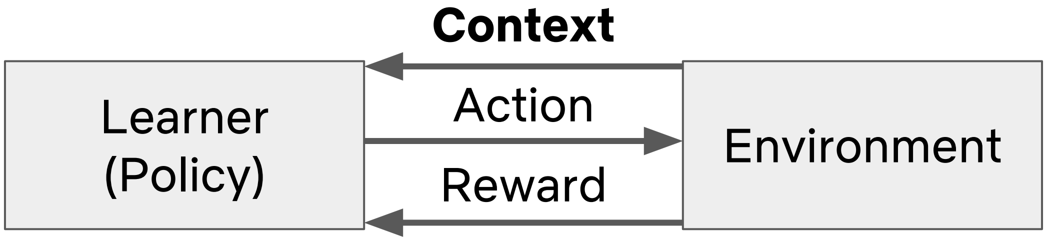
- Standard setting: **Stochastic and stationary**
- **Drifting**: Reward values change over time
- **Adversarial**: No assumptions on how rewards are generated
- **Continuous** action space
- **Infinite** set of actions
- **Varying** set of actions over time
- ...

What about **personalization**?

Contextual Bandits

- Let's make this harder!
- Slot machines where payout depends on context
- E.g. time of day, blinking light on slot machine, ...

Contextual Bandit



Each round:

- Environment provides **context** (feature) vector
- Learner chooses an **action** for context
- Environment provides a real-valued **reward** for action in context
- Learner updates to **maximize the cumulative reward**

Supervised Learning

Input: Features ($x \in \mathbb{R}^d$)

Output: Predicted label

Feedback: Actual label (y)

Contextual Bandits

Input: Context ($x \in \mathbb{R}^d$)

Output: Action ($a = \pi(x)$)

Feedback: Reward ($r \in \mathbb{R}$)

NETFLIX

Supervised Learning

Label

Cat ✗ Dog

Dog ✓ Dog

Dog ✓ Dog

Example Chihuahua images from [ImageNet]

Contextual Bandits

Reward

Cat ✗ 0

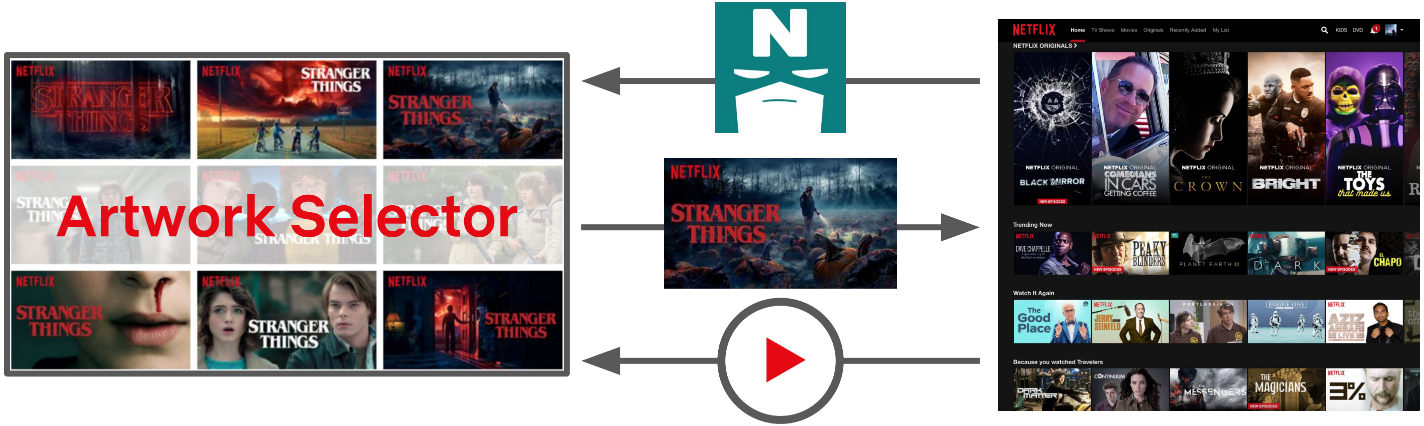
Fox ✗ 0

Seal ✗ 0

???

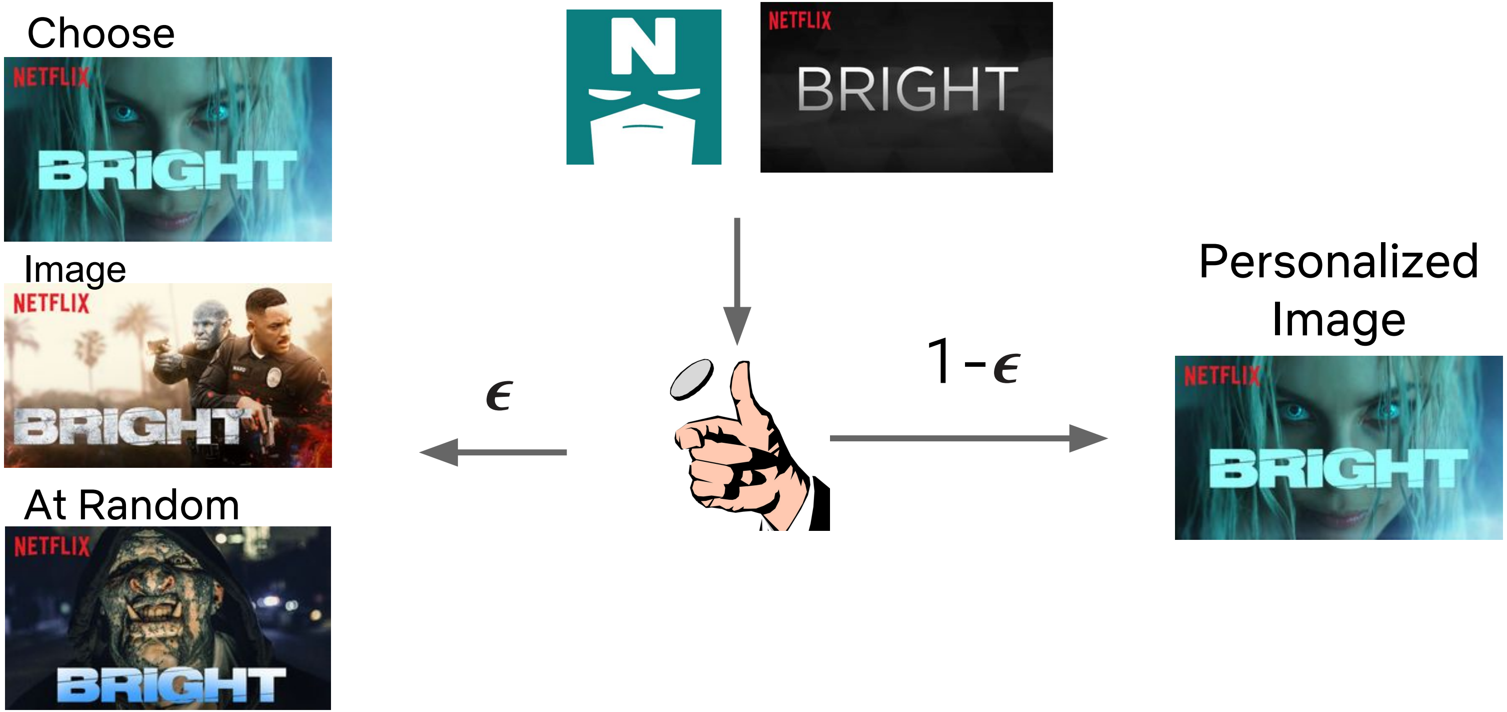
NETFLIX

Artwork Personalization as Contextual Bandit



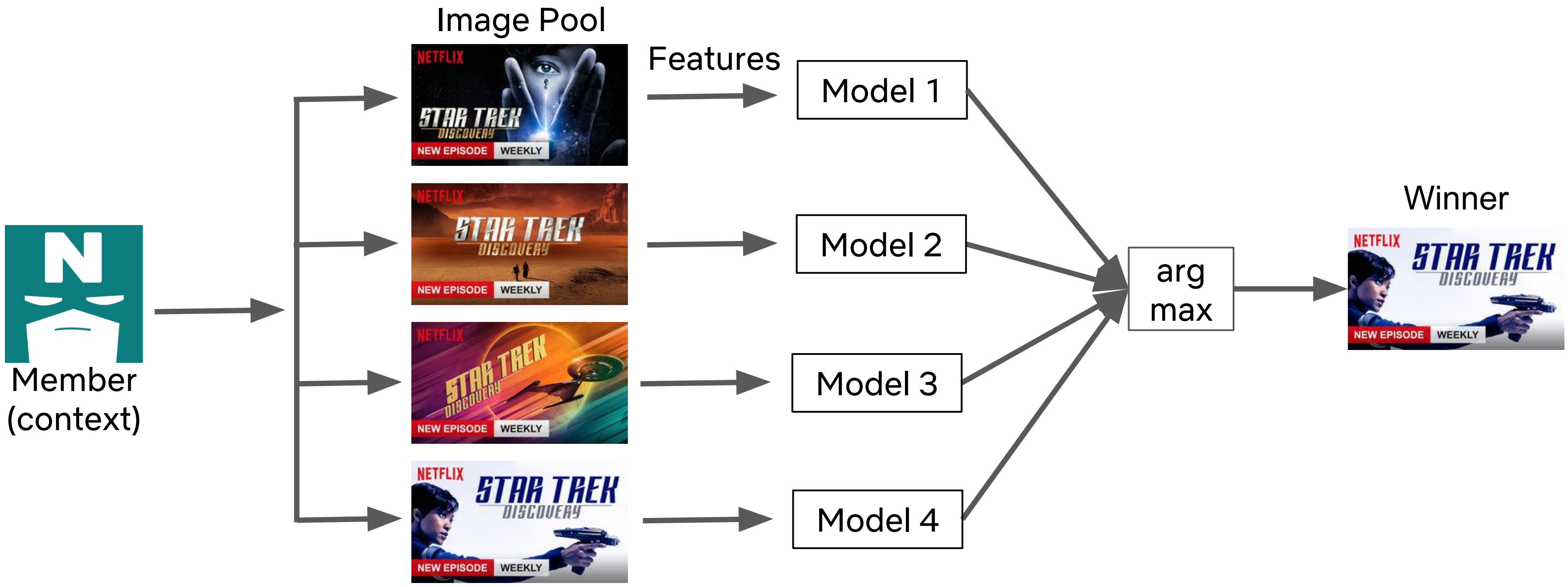
- **Context**: Member, device, page, etc.

Epsilon Greedy Example



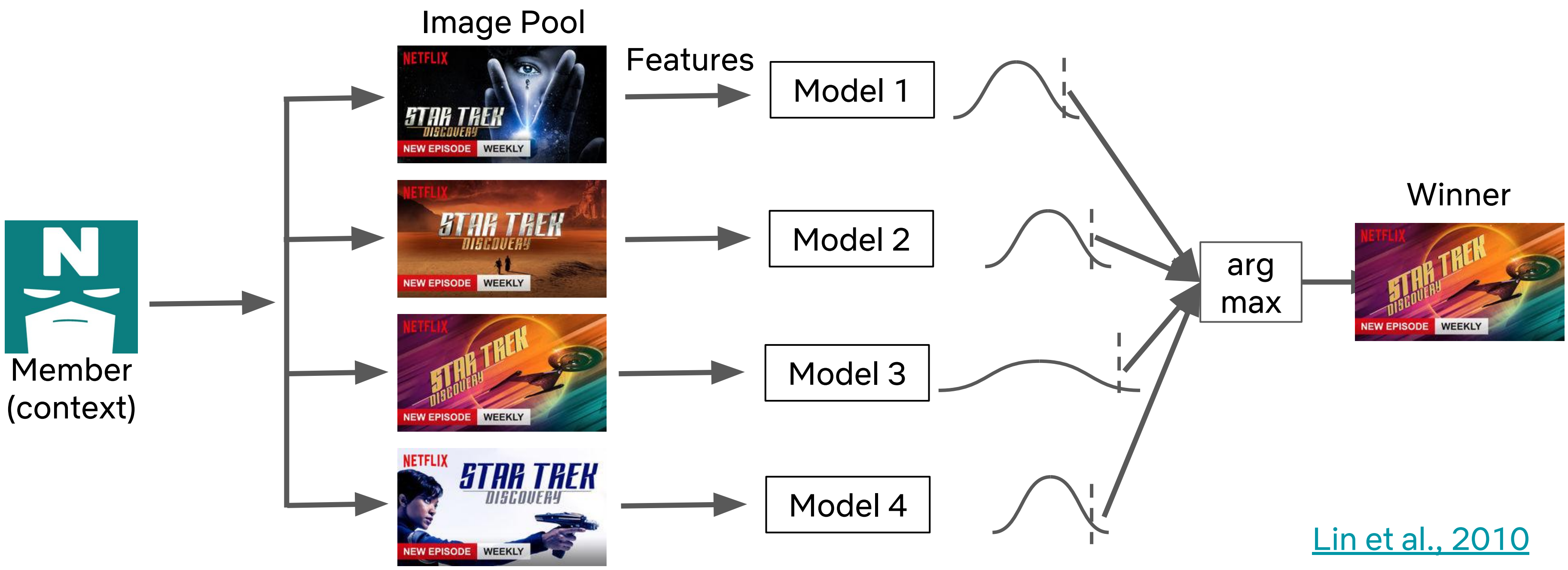
Greedy Policy Example

- Learn a supervised regression model per image to predict reward
- Pick image with highest predicted reward



LinUCB Example

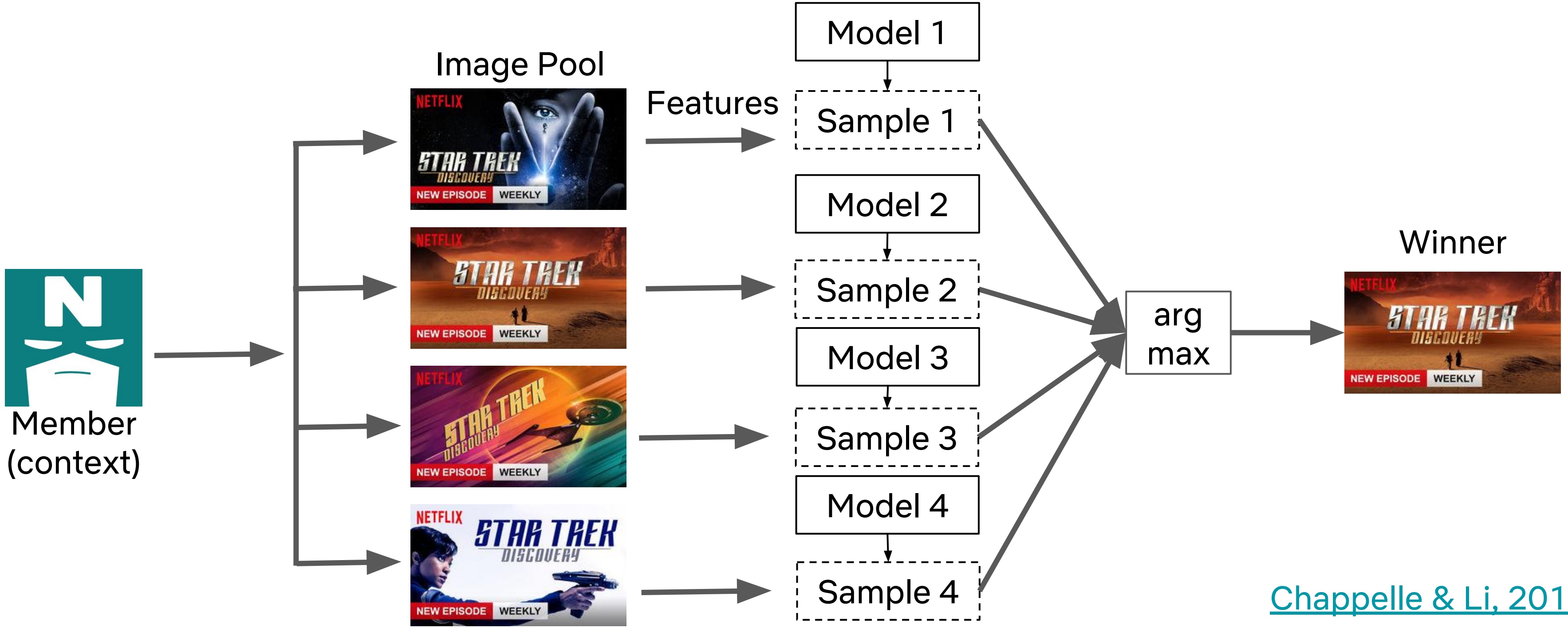
- Linear model to calculate uncertainty in reward estimate
- Choose image with highest α-percentile predicted reward value



Lin et al., 2010

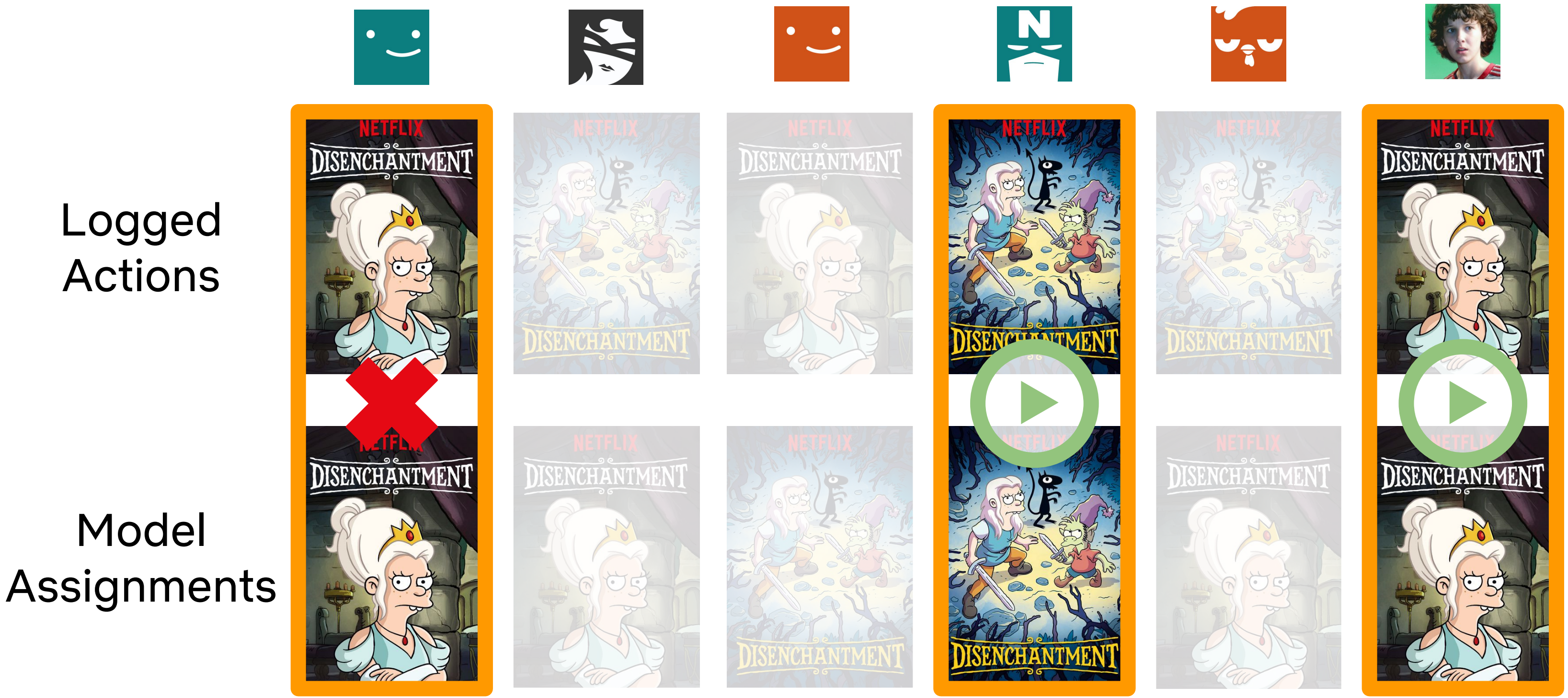
Thompson Sampling Example

- Learn distribution over model parameters (e.g. Bayesian Regression)
- Sample a model, evaluate features, take arg max



Chappelle & Li, 2011

Offline Metric: Replay



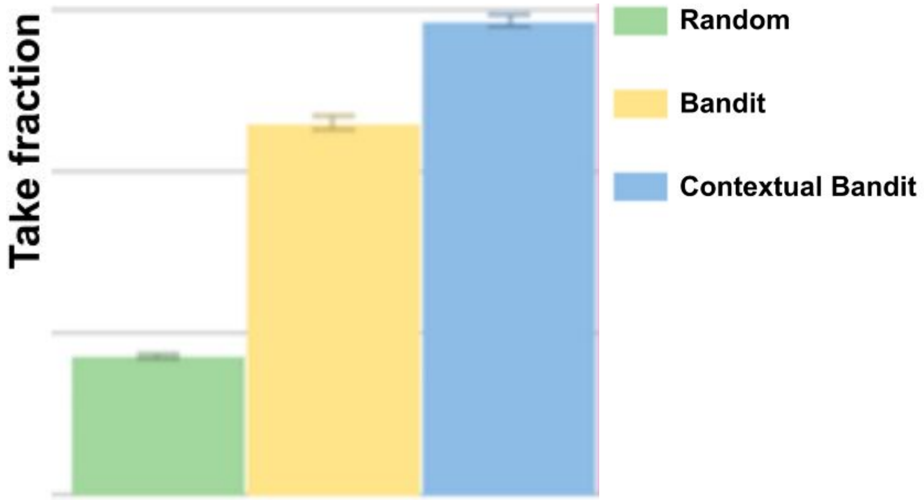
Offline Take Fraction: 2/3

[Li et al., 2011]

Replay

- Pros
  - **Unbiased** metric when using logged probabilities
  - Easy to compute
  - Rewards observed are real
- Cons
  - Requires a lot of data
  - **High variance** due if few matches
    - Techniques like Doubly-Robust estimation (Dudik, Langford & Li, 2011) can help

Offline Replay Results



Lift in Replay in the various algorithms as compared to the Random baseline

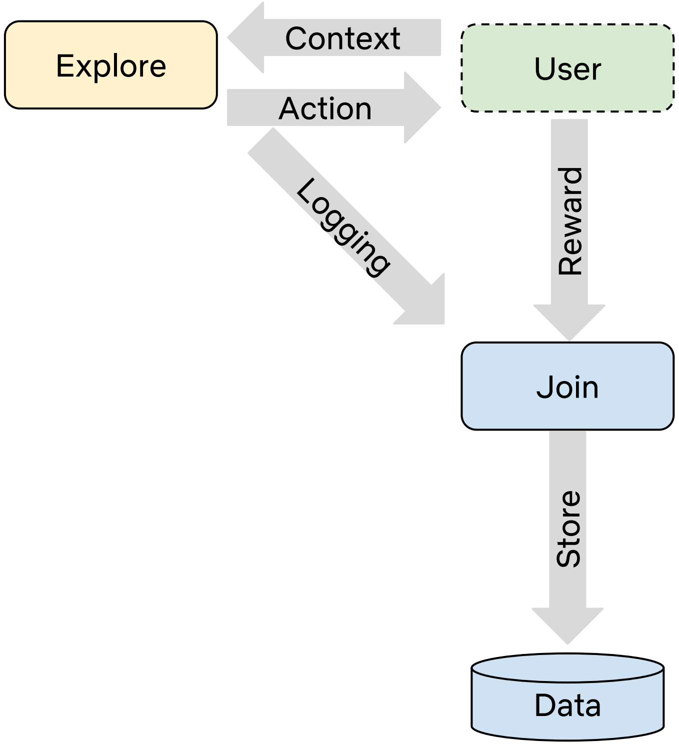
- Bandit finds good images
- Personalization is better
- Artwork variety matters
- Personalization wiggles around best images

Bandits in the Real World

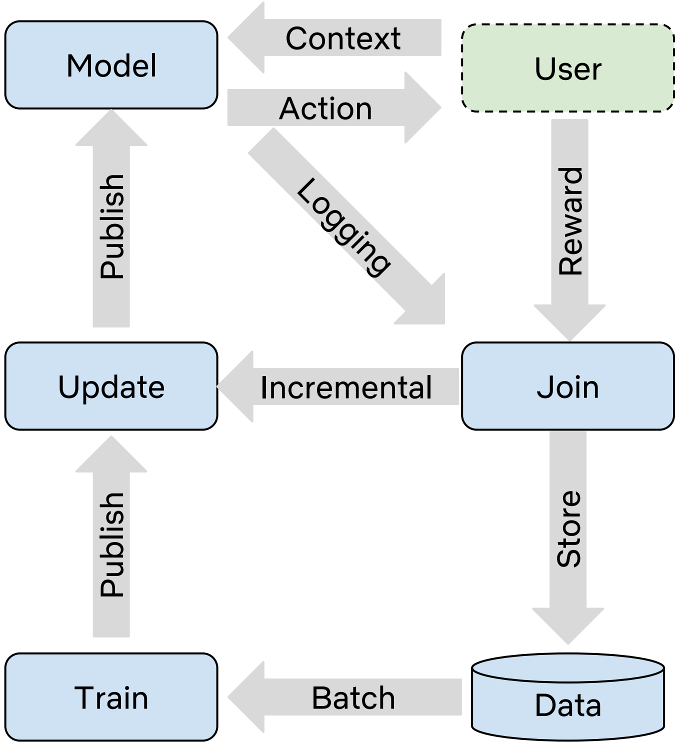
A/B testing Bandit Algorithms

- Getting started
  - Need data to learn
  - Warm-starting via batch learning from existing data
- Closing the feedback loop
  - Only exposing bandit to its own output
- Algorithm performance depends data volume
  - Need to be able to test bandits at large scale, head-to-head

Starting the Loop



Completing the Loop



Scale Challenges

- Need to serve an image for any title in the catalog
  - Calls from homepage, search, galleries, etc.
  - > 20M RPS at peak
- Existing UI code written assuming image lookup is fast
  - In memory map of video ID to URL
  - Want to insert Machine Learned model
  - Don't want a big rewrite across all UI code

Live Compute

Synchronous computation to choose image for title in response to a member request

Online Precompute

Asynchronous computation to choose image for title before request and stored in cache

Live Compute

Pros:

- Access to most **fresh** data
- Knowledge of **full context**
- Compute only what is necessary

Cons:

- **Strict** Service Level Agreements
  - Must respond quickly in **all cases**
  - Requires **high availability**
- Restricted to simple algorithms

See techblog for more details

Online Precompute

Pros:

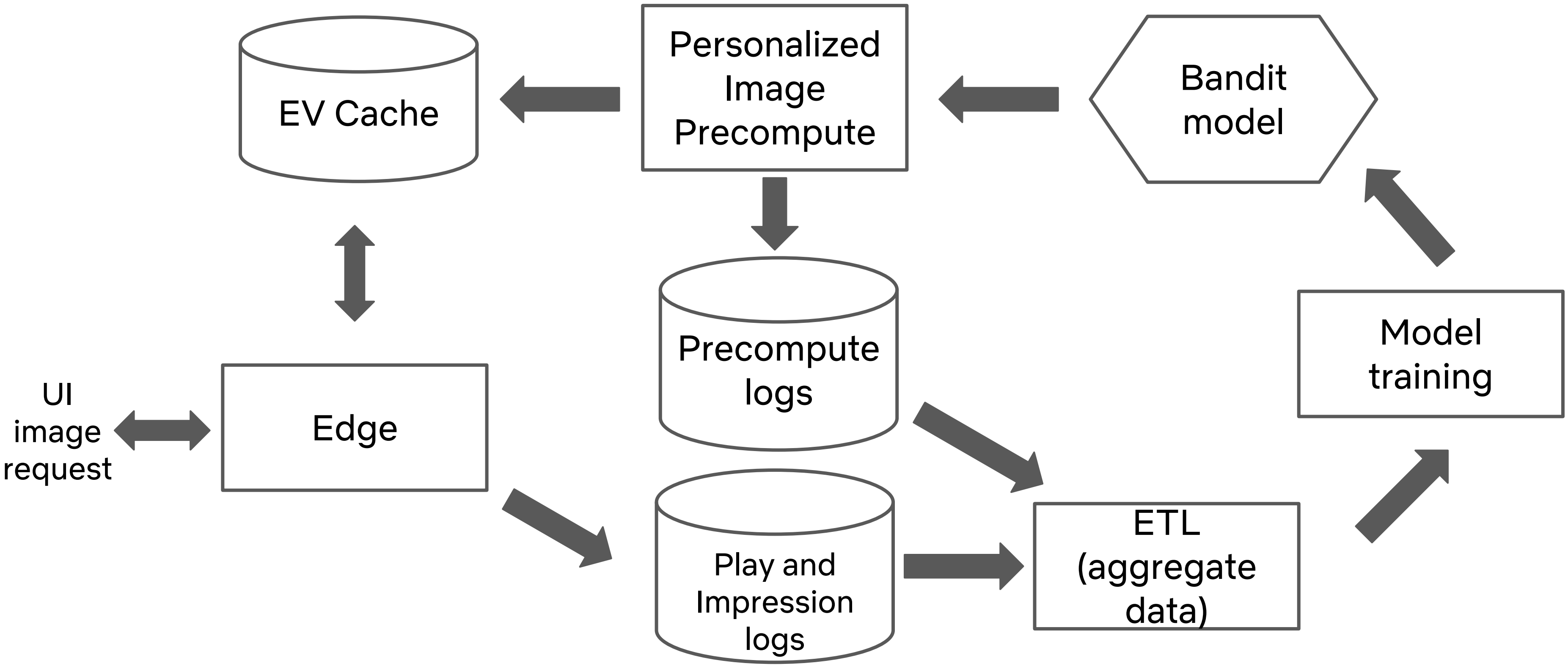
- Can handle **large data**
- Can run moderate complexity algorithms
- Can **average computational cost** across users
- Change from actions

Cons:

- Has some **delay**
- Done in **event context**
- **Extra compute** for users and items not served

NETFLIX

System Architecture



Precompute & Image Lookup

- Precompute
  - Run bandit for each title on each profile to choose personalized image
  - Store the title to image mapping in EVCache
- Image Lookup
  - Pull profile's image mapping from EVCache once per request

Logging & Reward

- Precompute Logging
  - Selected image
  - Exploration Probability
  - Candidate pool
  - Snapshot facts for feature generation
- Reward Logging
  - Image rendered in UI & if played
  - Precompute ID

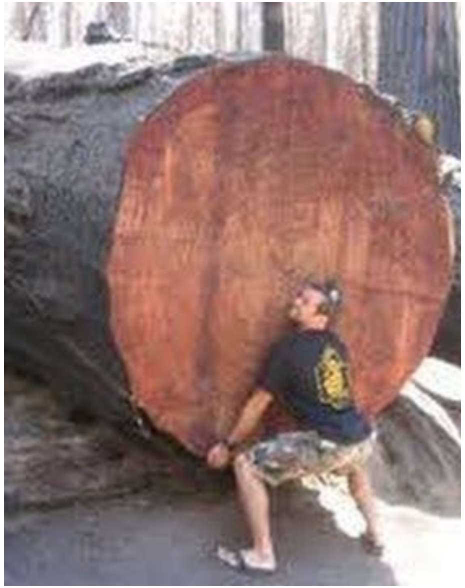


Image via YouTube

Feature Generation & Training

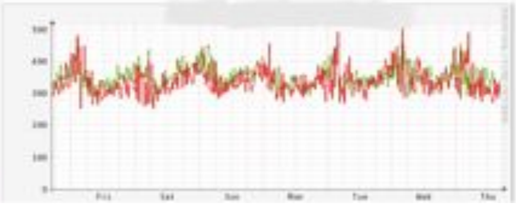
- **Join** rewards with snapshotted facts
- **Generate** features using DeLorean
  - Feature encoders are shared online and offline
- **Train** the model using Spark
- **Publish** model to production

DeLorean image by JMortonPhoto.com & OtoGodfrey.com

Monitoring and Resiliency

Track the **quality** of the model

- Compare prediction to actual behavior
- Online equivalents of offline metrics



Reserve a fraction of data for a simple policy (e.g. ϵ-greedy) to sanity check bandits

Graceful Degradation

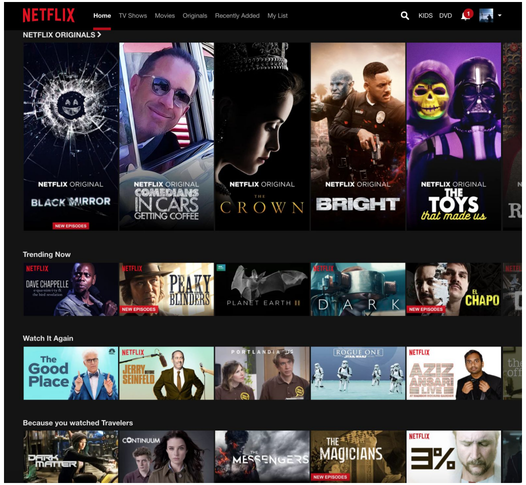
- Missing images greatly degrade the member experience
- Try to serve the best image possible

Personalized Selection	*Unpersonalized Fallback*	*Default Image (when all else fails)*

Does it work?

Online results

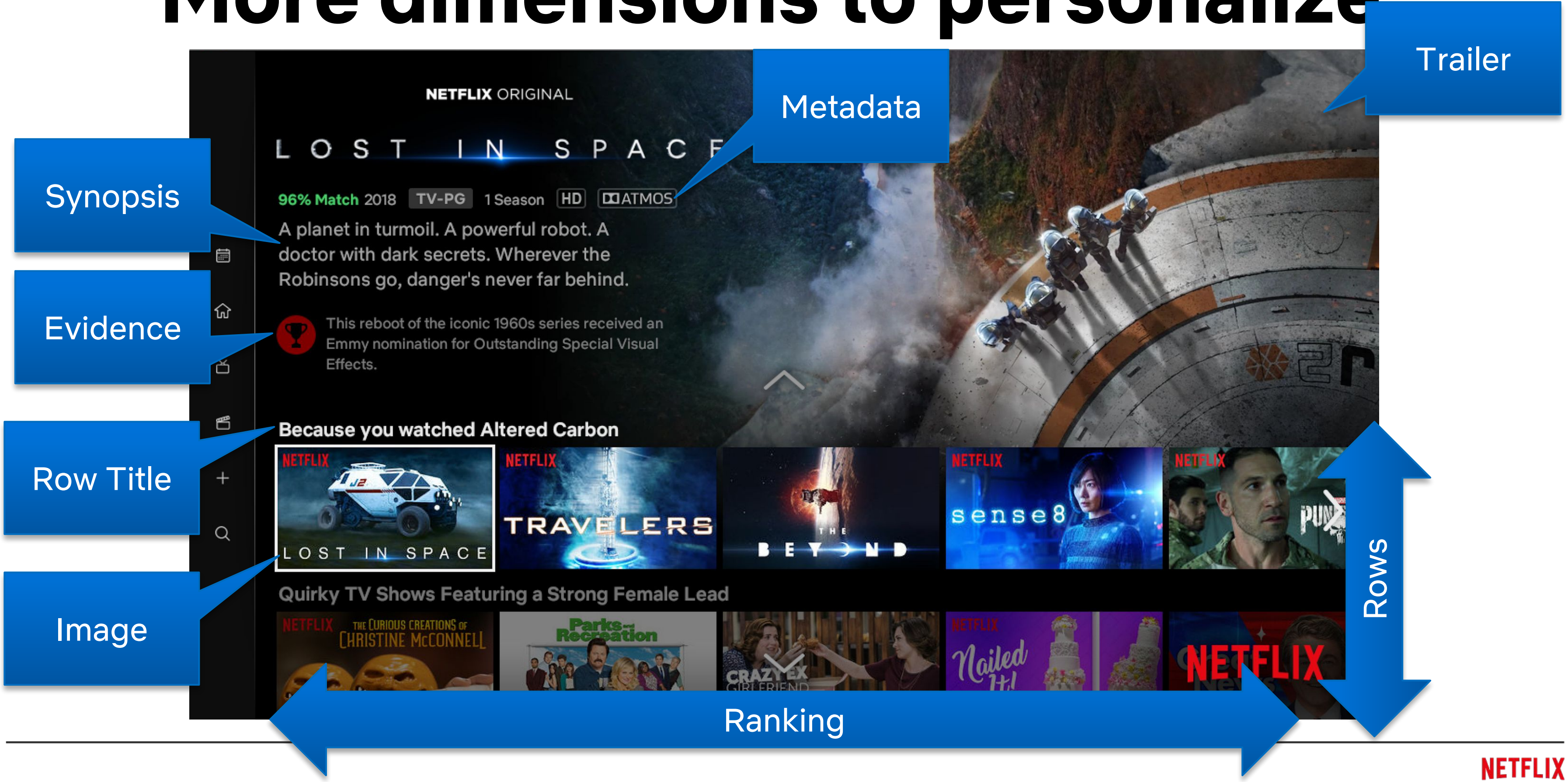
- A/B test: It works!
- Rolled out to our >130M member base
- Most beneficial for lesser known titles
- Competition between titles for attention leads to compression of offline metrics



More details in our [blog post]

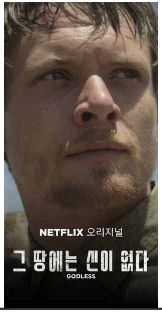
Future Work

More dimensions to personalize



Automatic image selection

- Generating new artwork is costly and time consuming
- Can we predict performance from raw image?



Artwork selection orchestration

- Neighboring image selection influences result

Example: Stand-up comedy

Row A
(microphones)

Row B
(more variety)

Long-term Reward: Road to Reinforcement Learning

- RL involves multiple actions and delayed reward
- Useful to maximize user long-term joy?

NETFLIX

Thank you

@JustinBasilico

NETFLIX