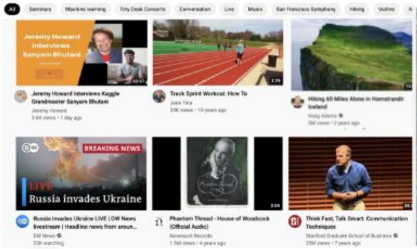
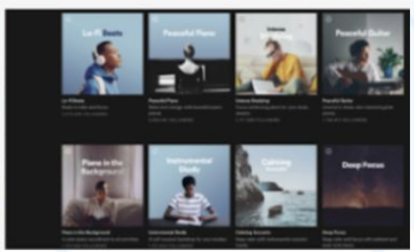
Hands-on Reinforcement Learning for RecSys – from Bandits to Offline RL with Ray RLlib

Kourosh Hakhamaneshi - kourosh@anyscale.com
Christy Bergman - christy@anyscale.com





Few Important URLs

Keep these URLs open in your browser tabs

→ **GitHub: https://bit.ly/rllib_recsys_2022_github**
→ Q & A Doc: https://bit.ly/rllib_recsys_2022-qa
→ Logins+passwords: https://bit.ly/rlib_recsys-logins
→ Anyscale: console.anyscale.com
→ Tutorial Survey: https://bit.ly/rlib_recsys_2022

Few Important URLs

Keep these URLs open in your browser tabs

→ GitHub: https://bit.ly/rllib_recsys_2022_github
→ **Q & A Doc: https://bit.ly/rllib_recsys_2022-qa**
→ Logins+passwords: https://bit.ly/rlib_recsys-logins
→ Anyscale: console.anyscale.com
→ Tutorial Survey: https://bit.ly/rlib_recsys_2022

Few Important URLs

Keep these URLs open in your browser tabs

→ GitHub: https://bit.ly/rllib_recsys_2022_github
→ Q & A Doc: https://bit.ly/rllib_recsys_2022-qa
→ **Logins+passwords: https://bit.ly/rlib_recsys-logins**
→ **Anyscale: console.anyscale.com**
→ Tutorial Survey: https://bit.ly/rlib_recsys_2022

Few Important URLs

Keep these URLs open in your browser tabs

→ GitHub: https://bit.ly/rllib_recsys_2022_github
→ Q & A Doc: https://bit.ly/rllib_recsys_2022-qa
→ Logins+passwords: https://bit.ly/rlib_recsys-logins
→ Anyscale: console.anyscale.com
→ **Tutorial Survey: https://bit.ly/rlib_recsys_2022**

$whoami (Christy)

→ AI/ML DevAdvocate @Anyscale.
→ Previously: AI/ML Solutions Architect at AWS, before that data scientist real-time fraud detection

$whoami (Kourosh)

→ ML Engineer working on RL and RLlib @Anyscale.
→ Previously: PhD student at UC Berkeley working on RL in Robotics and design optimization

RL Team @ Anyscale

Sven

Jun

Avnish

Artur

Kourosh

Christy
(devAdvocate)

Anyscale

Who we are: Original creators of Ray, a unified framework for scalable, distributed computing. Part of that framework are our libraries for ML and data processing.

What we do: Scalable compute for AI and Python

Why we do it: Scaling is a necessity, scaling is hard; make distributed computing easy and simple for all developers.

Some of RLlib's Industry Users

Overview of the tutorial

- → Brief intro RL
- → Brief intro RecSys
  - \+ Traditional Approaches
  - \+ Defining RecSys as an RL problem
- → Online RL vs Offline RL
- → Hands-on coding with python notebooks and scripts

Goals – Understand:

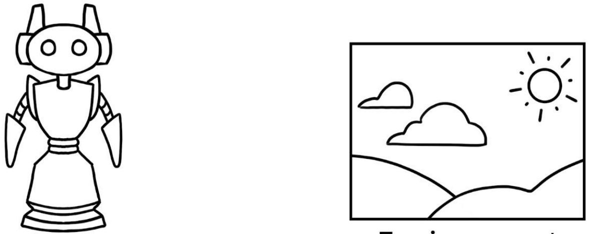
- ➢ What are the advantages of using RL in RecSys?
- ➢ What are the pros and cons of offline RL in practical scenarios?

Overview of the tutorial

→ 4 min: Welcome
→ 5 min: Very brief intro RL
→ 5 min: Very brief intro RecSys
  + Machine learning (ML) approach
  + Challenges with current ML approach
  + Map RecSys problem into MDP for RL
→ 5 min: Intro Online RL vs Offline RL
→ 1 hour: Hands-on with Google Colab
  + 15min: Introduction to the environment
  + 10min: Run baselines, bandit, and RL algorithm
  + 5min: Conclusion so far TODO ADD slide with results
  + 10min: Run offline RL on expert, random, greedy data
  + 5min: Conclusion so far TODO ADD slide with results
  + 5min: Deploy a policy to production using Ray Serve

Brief intro RL

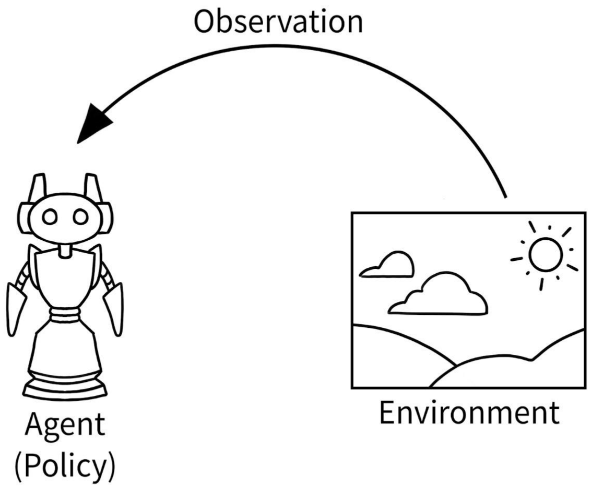
Conversation between an agent and an environment.



Agent
(Policy)

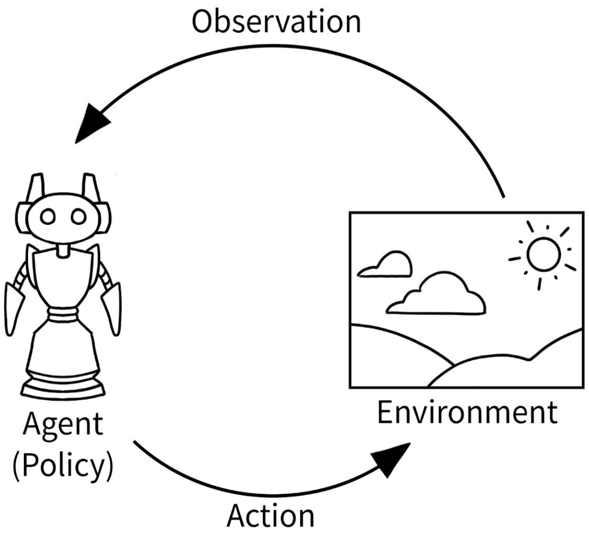
Environment

Brief intro RL



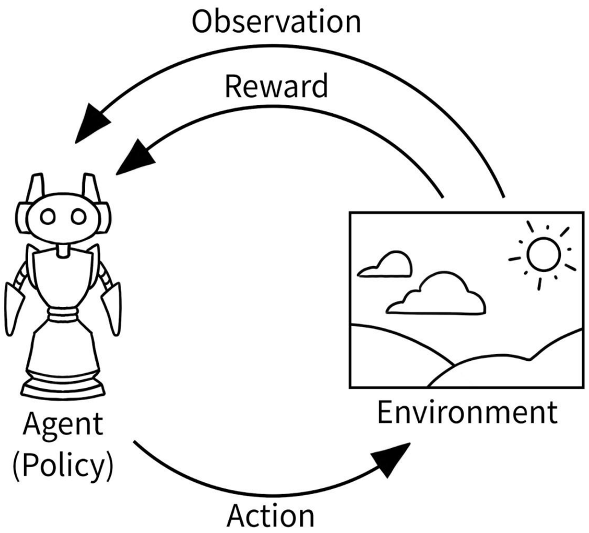
Conversation between an agent and an environment.

Brief intro RL



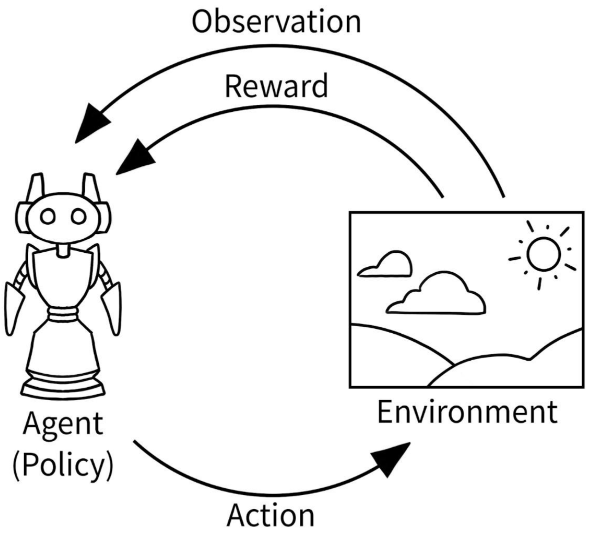
Conversation between an agent and an environment.

Brief intro RL



Conversation between an agent and an environment.

Brief intro RL



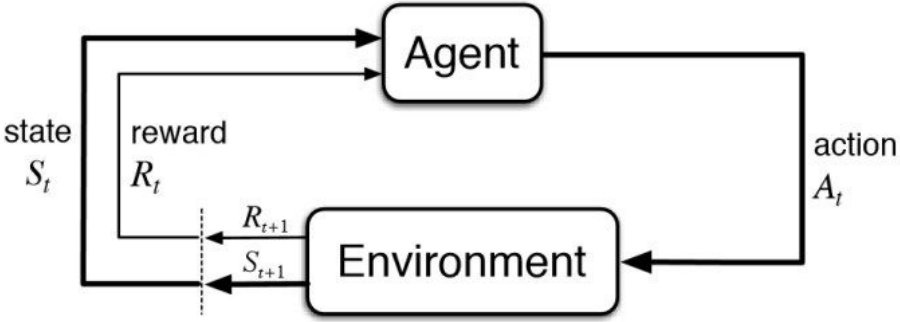
Conversation between an agent and an environment.

Learning objectives:

- Maximize sum of rewards.
- Learn from delayed reward.
- Proper exploration to maximally learn

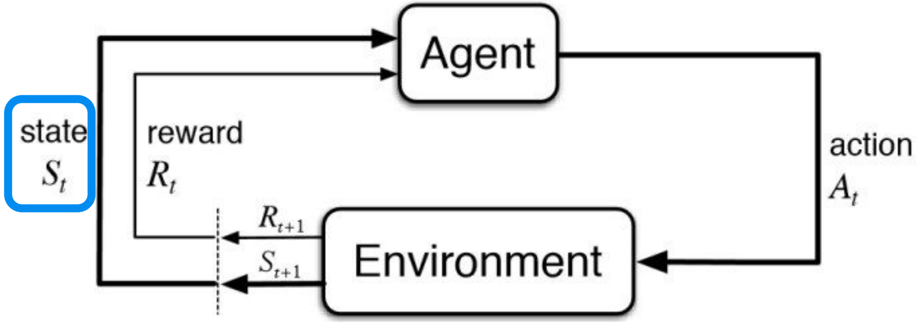
Brief intro RL

$$(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$$



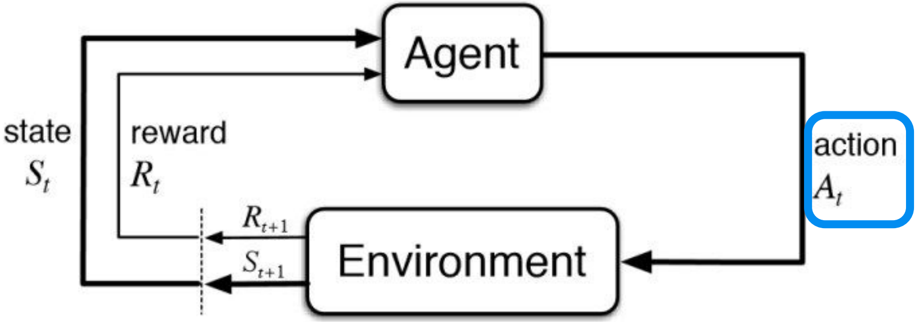
Brief intro RL

$$\left(\boxed{\mathcal{S},} \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma\right)$$

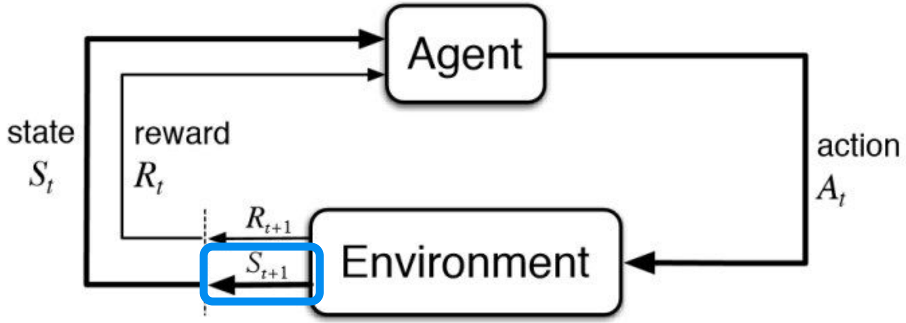


Brief intro RL

$$(\mathcal{S}, \boxed{\mathcal{A},} \mathcal{P}, \mathcal{R}, \gamma)$$



Brief intro RL

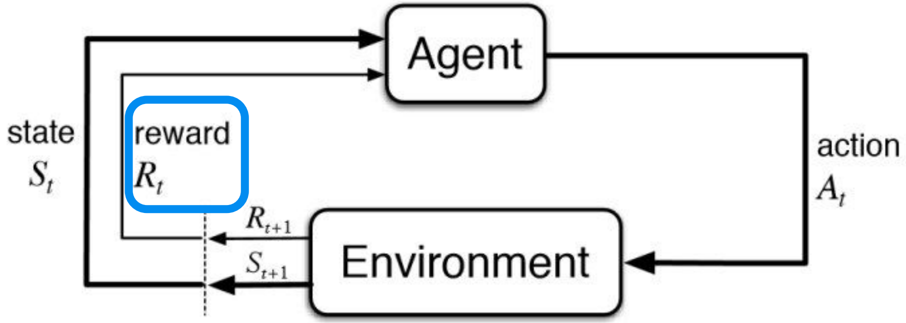


$$(\mathcal{S}, \mathcal{A}, \boxed{\mathcal{P}}, \mathcal{R}, \gamma)$$

$$S_0 \sim \mathcal{P}(.)$$

$$S_{t+1} \sim \mathcal{P}(.|S_t, A_t)$$

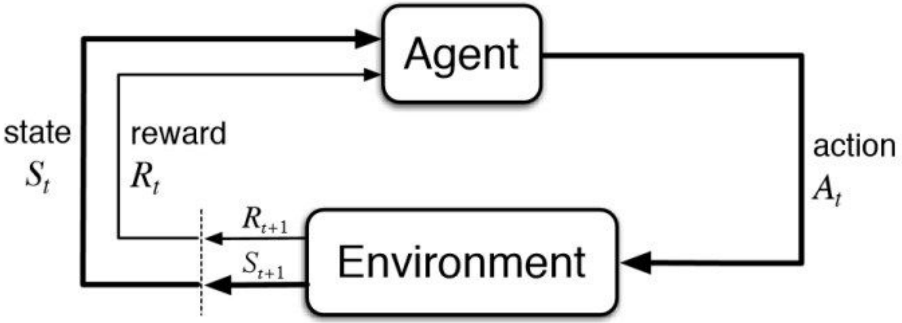
Brief intro RL



$$(\mathcal{S}, \mathcal{A}, \mathcal{P}, \boxed{\mathcal{R},} \gamma)$$

$$R_t = \mathcal{R}(S_t, A_t)$$

Brief intro RL

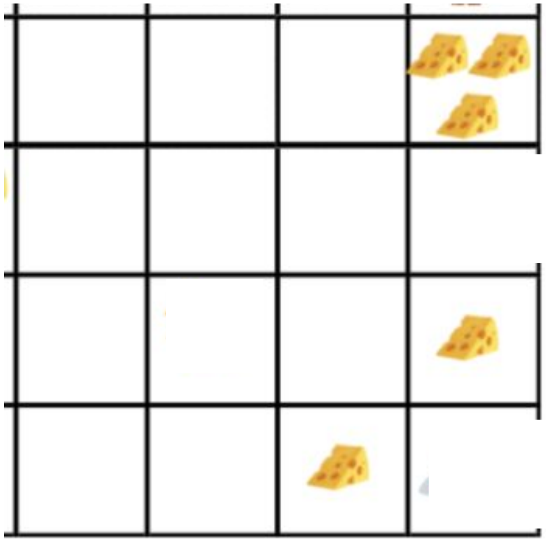


$$(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \boxed{\gamma})$$

$$R(\tau) = \sum_t \gamma^t R_t$$

Discount factor γ in RL

- If γ **= 0**, the algorithm considers **1-step rewards only**.
- If γ **= 1,** the algorithm considers all future rewards equally.

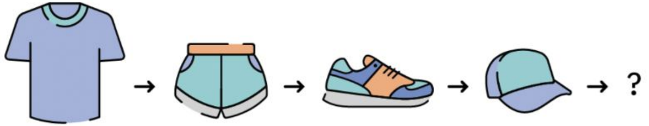


Brief intro RecSys

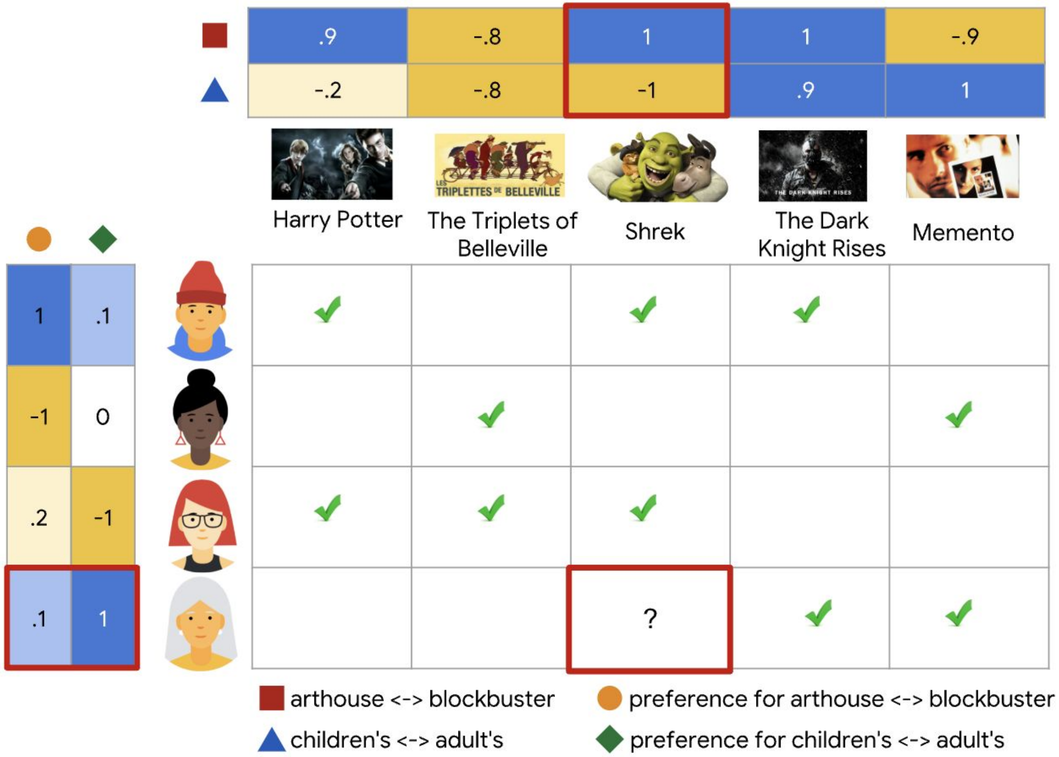
Companies want to recommend content.

ML: Pointwise recommendations.

RL: Combine pointwise recommendations with session based data.



Brief intro RecSys ML



Credit: https://developers.google.com/machine-learning/

Challenges with traditional ML in RecSys

- Traditional ML (collaborative filtering) models are **static with respect to time**.
- This type of model **ignores time order** in which users did actions.

Challenges with traditional ML in RecSys

- Traditional ML (collaborative filtering) models are **static with respect to time**.
    - Ignores the **sequence of interactions** with a given user.

Challenges with traditional ML in RecSys

- Traditional ML (collaborative filtering) models are **static with respect to time**.
  - Ignores the **sequence of interactions** with a given user.
- Static models can be:
  - Too short-sighted and **miss out on Long-term, delayed rewards**

Challenges with traditional ML in RecSys

- Traditional ML (collaborative filtering) models are **static with respect to time**.
  - Ignores the **sequence of interactions** with a given user.
- Static models can be:
  - Too short-sighted and **miss out on Long-term, delayed rewards**
  - **Overlook important and changing user intents** or business conditions such as seasonality or promotional campaigns

New way: RL in RecSys

- By taking each user's session history as a sequence of decisions, **the RecSys problem can be converted into a sequential decision-making problem**.
  - $Pr[R(t+1)=r_t \mid A(t)=a_t, S(t)=s_t, A(t-1)=a_{t-1}, \ldots S(0)=s_0]$
- A stochastic process is a **Markov Decision Process (MDP)** if the values at time t depend only on the values at time t-1.
  - $Q_\pi(s,a) = E_\pi\left[\sum_{j=0}^{T} \gamma^j r_{t+j+1} \middle| S_t = s, A_t = a\right]$
- **RL has become the de-facto ML approach for solving MDPs.**

New way: RL in RecSys

- By taking each user's session history as a sequence of decisions, **the RecSys problem can be converted into a sequential decision-making problem**.

New way: RL in RecSys

- By taking each user's session history as a sequence of decisions, **the RecSys problem can be converted into a sequential decision-making problem**.

  States: item features, user feature, history of interactions

New way: RL in RecSys

- By taking each user's session history as a sequence of decisions, **the RecSys problem can be converted into a sequential decision-making problem**.

  States: item features, user feature, history of interactions

  Actions: the items to recommend

New way: RL in RecSys

- By taking each user's session history as a sequence of decisions, **the RecSys problem can be converted into a sequential decision-making problem**.

  States: item features, user feature, history of interactions

  Actions: the items to recommend

  Reward: long term satisfaction (explicit or implicit)

New way: RL in RecSys

- By taking each user's session history as a sequence of decisions, **the RecSys problem can be converted into a sequential decision-making problem**.

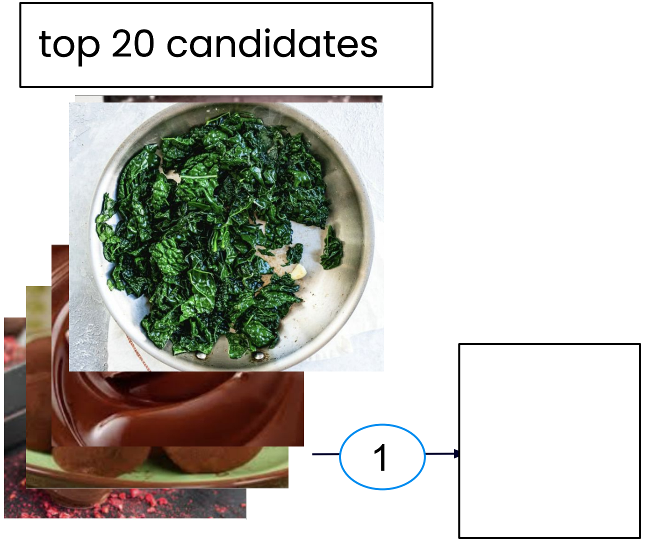
  States: item features, user feature, history of interactions

  Actions: the items to recommend

  Reward: long term satisfaction (explicit or implicit)

  Gamma: 0 (bandits) or 1 (RL)

RL Environment: Delayed Rewards & Long Term Satisfaction (LTS)



RL Environment: Delayed Rewards & Long Term Satisfaction (LTS)

top 20 candidates

1

2

Recommendation
1 (action)

RL Environment:
Delayed Rewards &
Long Term Satisfaction (LTS)

top 20 candidates

1

2

Recommendation
1 (action)

User decides to click or not

3

States, actions, rewards updated.
Policy model is retrained.
Best next action is sampled

4

Repeat until user quits
or end of time limit

$$max_\pi \left(\sum_{j=0}^{T} \gamma^j \hat{R}_j\right)$$

RL Environment: Delayed Rewards & Long Term Satisfaction (LTS)

top 20 candidates

Recommendation
1 (action)

User decides to click or not

3

States, actions, rewards updated.
Policy model is retrained.
Best next action is sampled

4

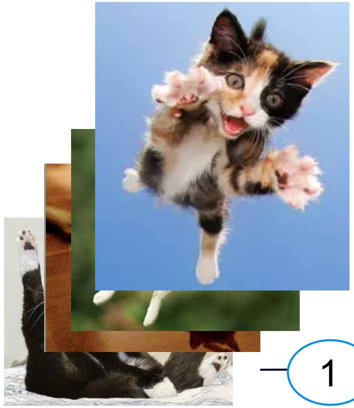
$max_{\pi} \left(\sum_{j=0}^{T} \gamma^{j} \hat{R}_{j}\right)$

Repeat until user quits
or end of time limit

5

RL Environment:
Delayed Rewards &
Long Term Satisfaction (LTS)

top 20 candidates



1

2

Recommendation
1 (action)

User decides to click or not

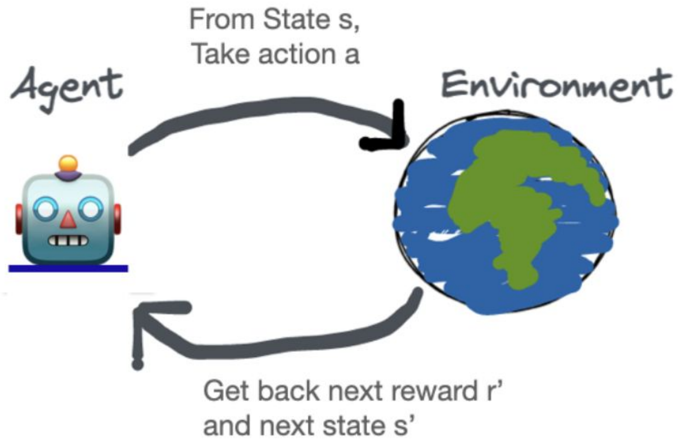
3

States, actions, rewards updated.
Policy model is retrained.
Best next action is sampled

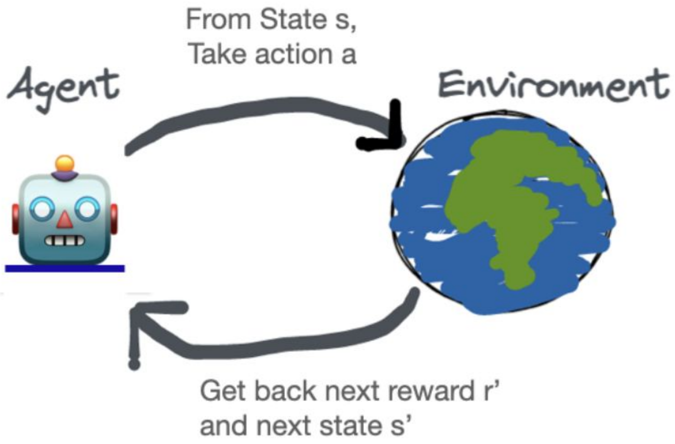
$$max_{\pi}\left(\sum_{j=0}^{T}\gamma^{j}\hat{R}_{j}\right)$$

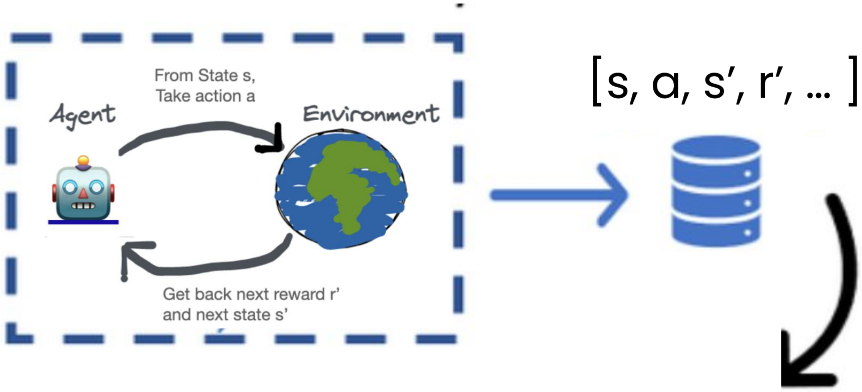
Repeat until user quits
or end of time limit

Online RL vs Offline RL

Online RL	Batch / Offline RL
	

Online RL vs Offline RL

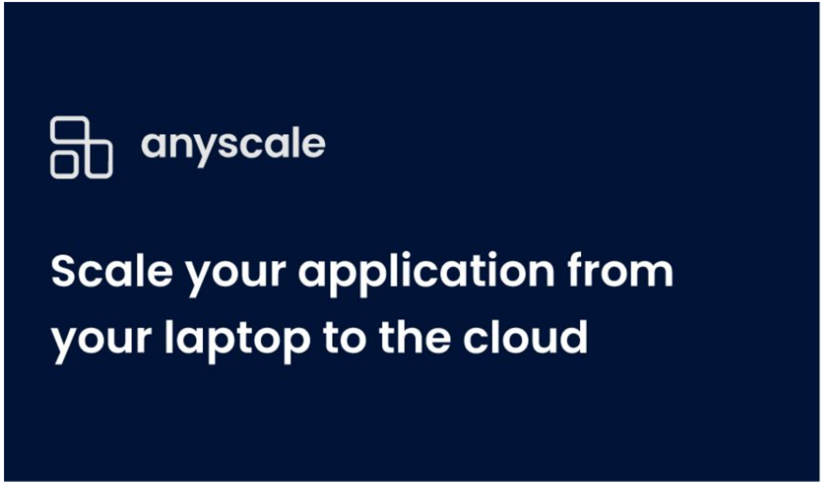
<table>
<tr><th>Online RL</th><th>Batch / Offline RL</th></tr>
<tr><td>


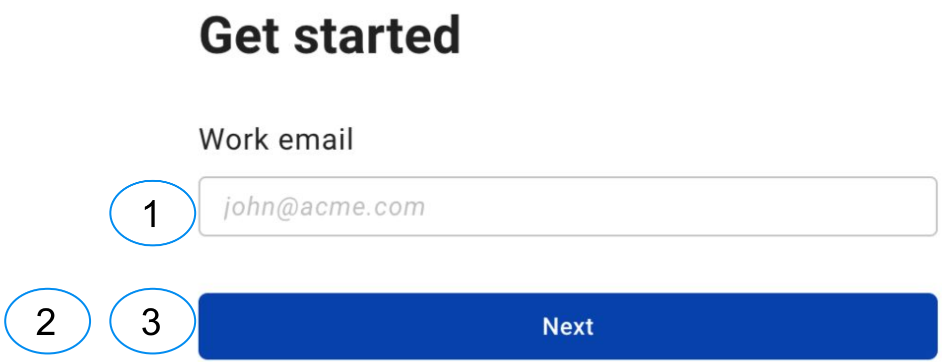
</td><td>


</td></tr>
</table>

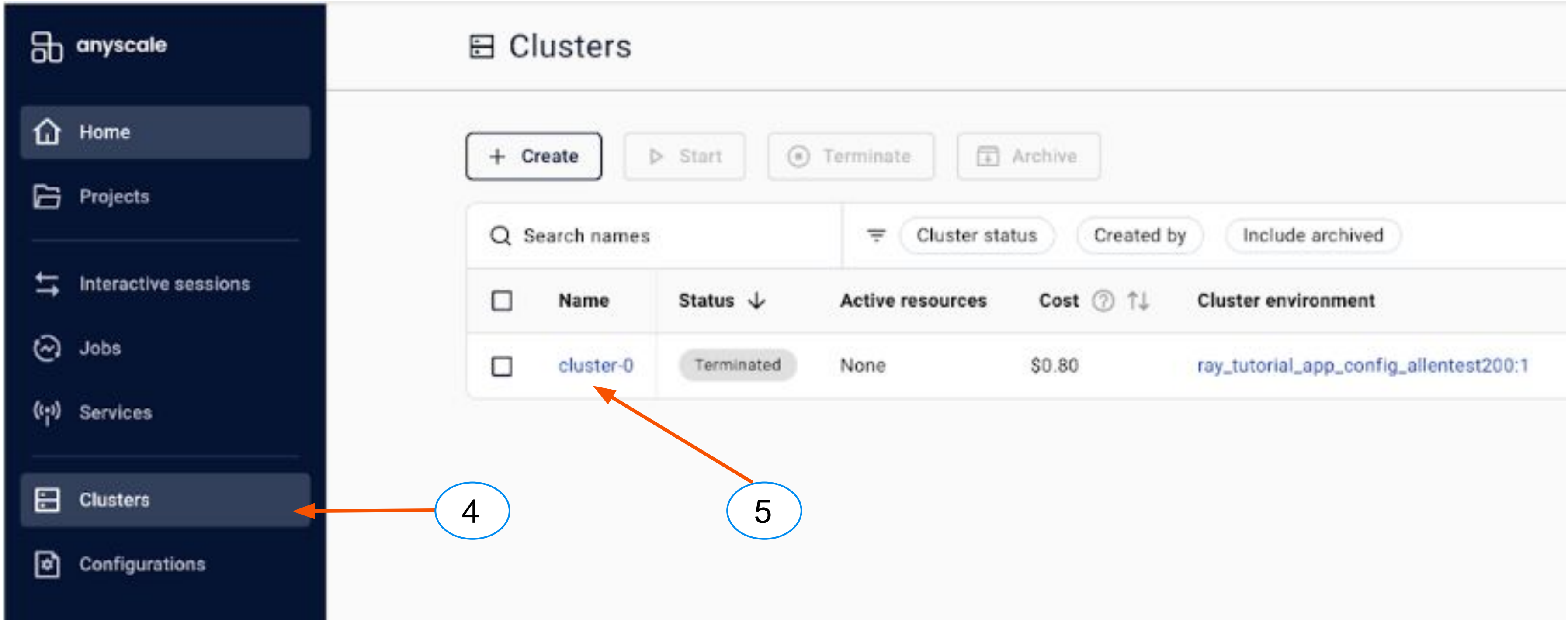
Your Anyscale Cluster

- → Claim username/password at https://bit.ly/rlib_recsys-logins
  - + Update the "Status" column to "Available" or "Claimed"
- → Go to Console: http://console.anyscale.com/
- → Enter username (for the email) and password

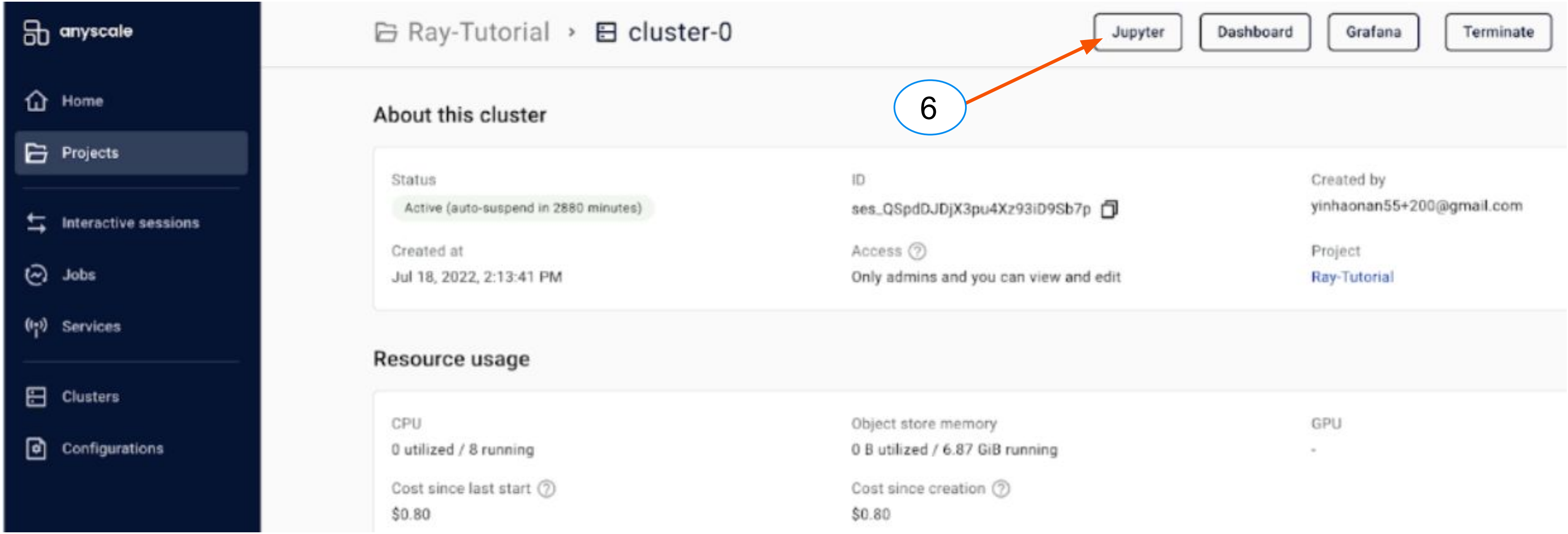




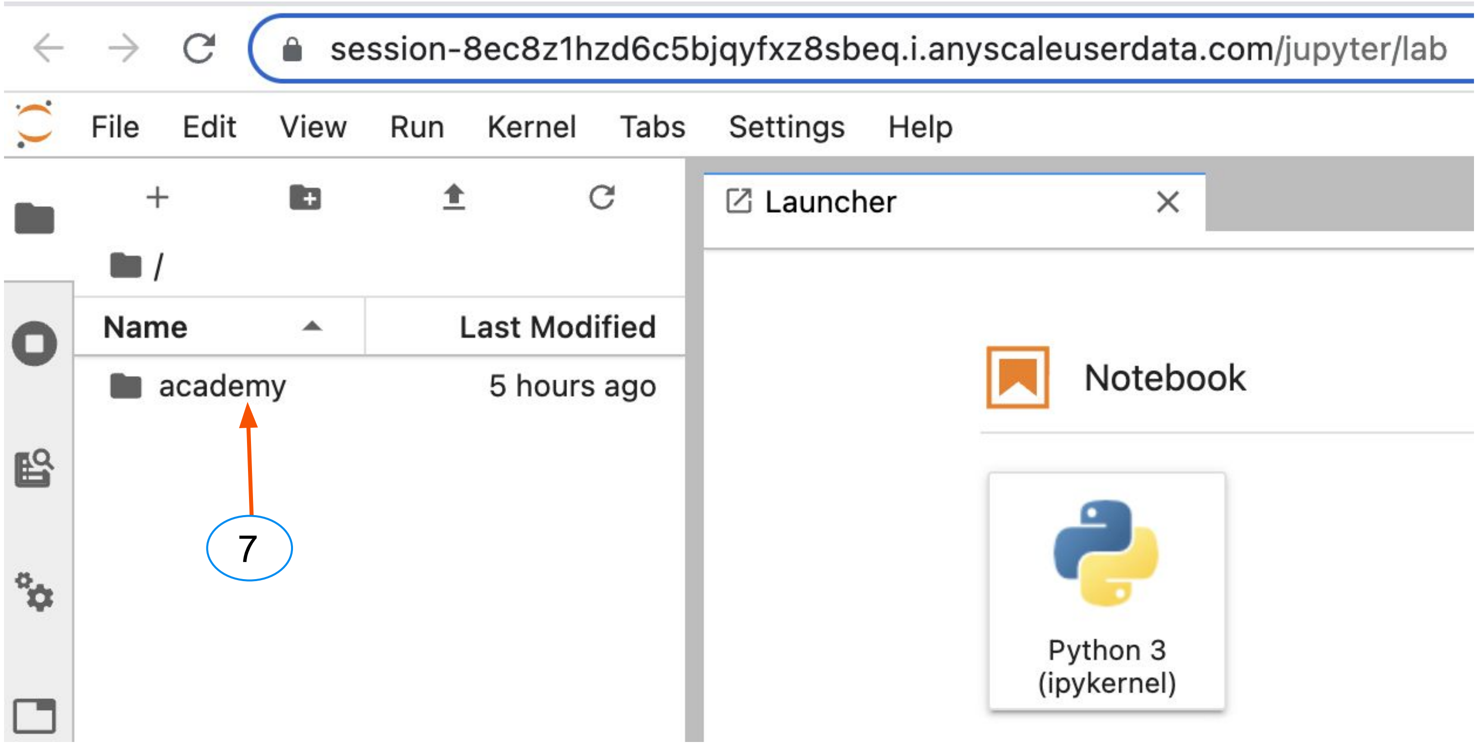
Your Anyscale Cluster



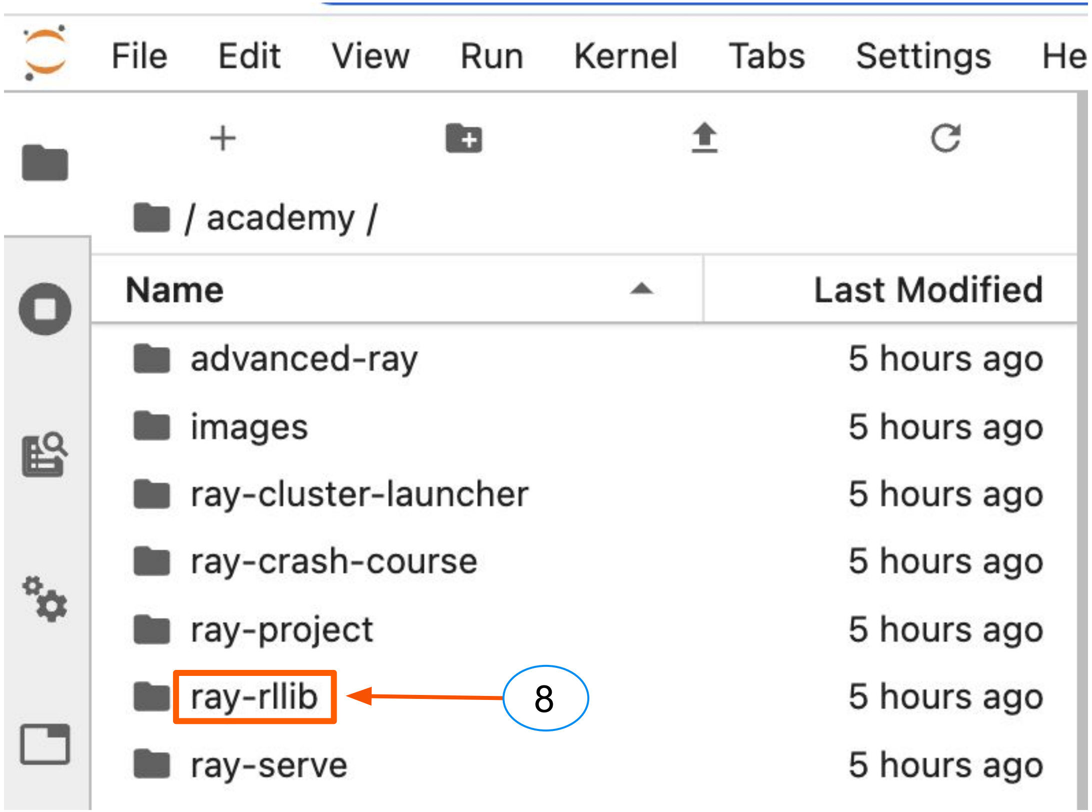
Your Anyscale Cluster



Your Anyscale Cluster

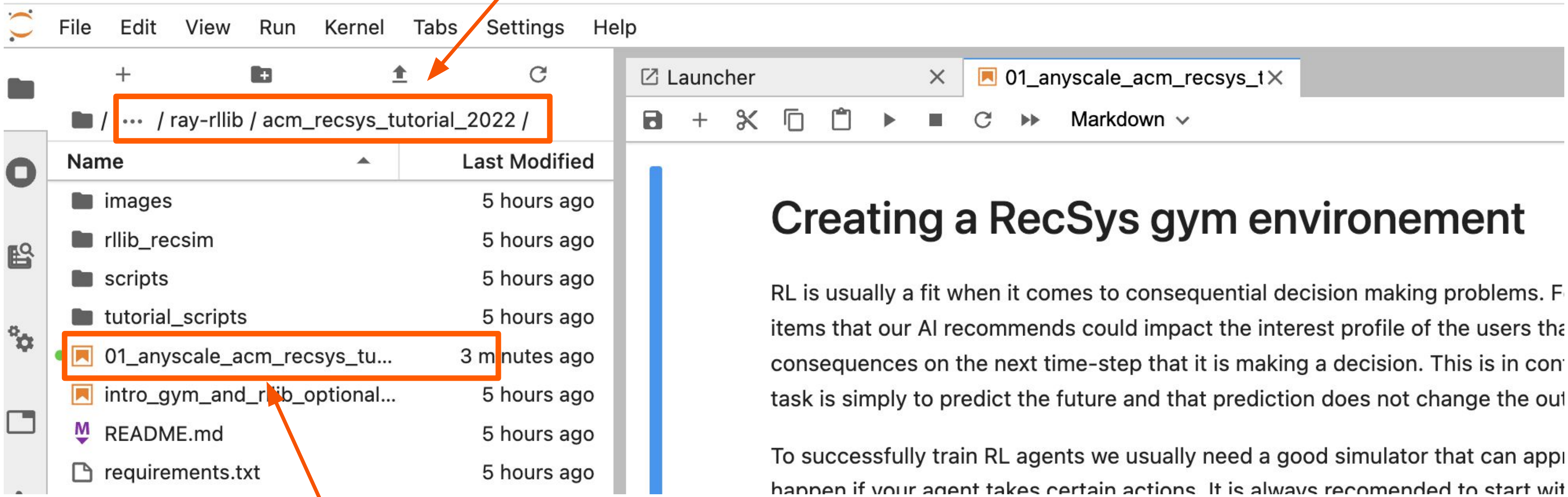


Your Anyscale Cluster



Your Anyscale Cluster

9 Navigate to “acm_recsys_tutorial_2022”



10 Open notebook “01_...”

Thank you.

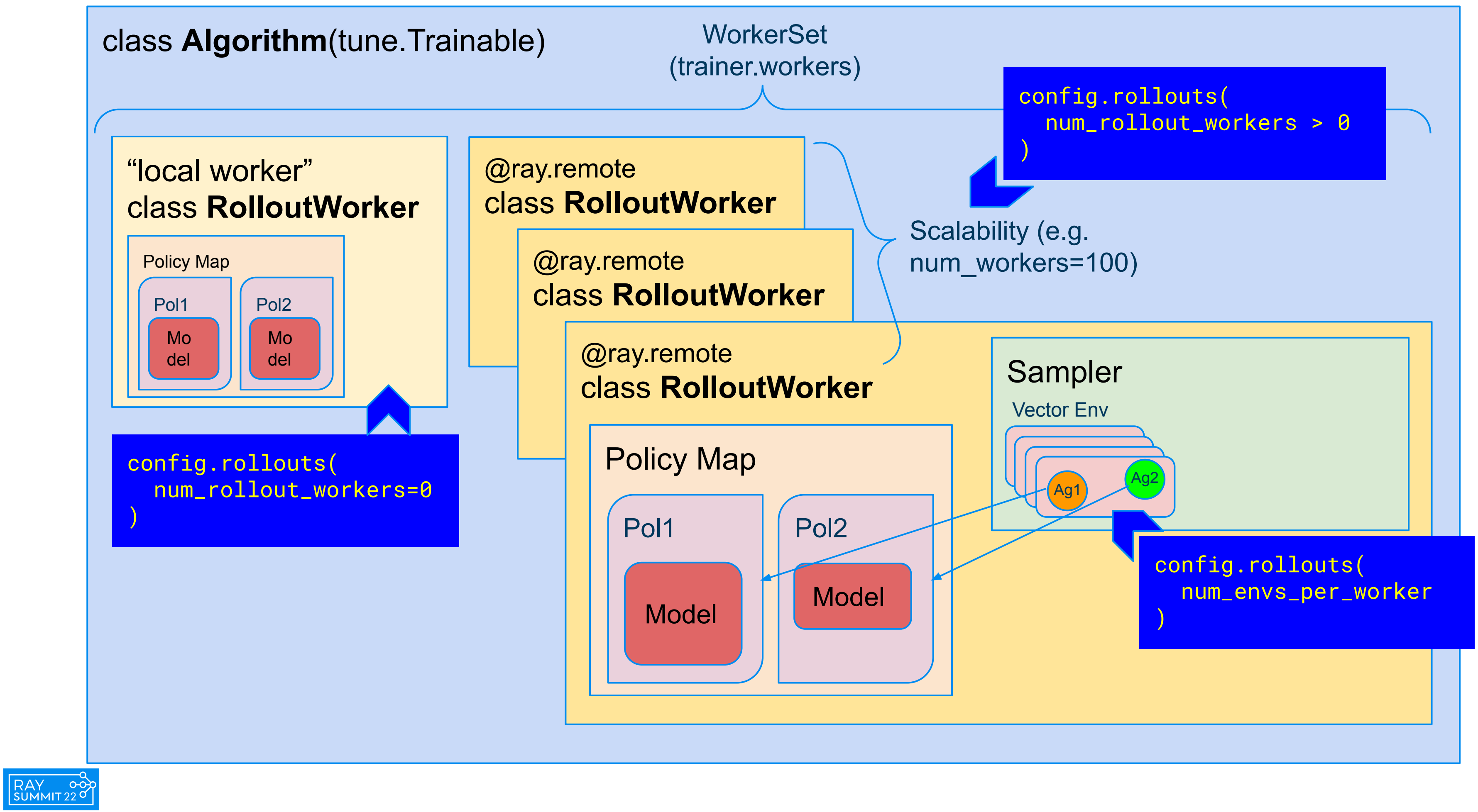
We would love to connect with you!

Twitter - @anyscalecompute | @raydistributed

GitHub - https://github.com/ray-project/ray

Slack - https://www.ray.io/community

Discuss - https://discuss.ray.io/


class Algorithm(tune.Trainable)
WorkerSet
(trainer.workers)
config.rollouts(
num_rollout_workers > 0
)
"local worker"
class RolloutWorker
Policy Map
Pol1
Mo
del
Pol2
Mo
del
@ray.remote
class RolloutWorker
@ray.remote
class RolloutWorker
Scalability (e.g.
num_workers=100)
@ray.remote
class RolloutWorker
Sampler
Vector Env
Ag1
Ag2
Policy Map
Pol1
Model
Pol2
Model
config.rollouts(
num_rollout_workers=0
)
config.rollouts(
num_envs_per_worker
)
RAY
SUMMIT 22