Reinforcement Learning: A Visual Guide

Teaching Computers to Make Decisions

Prof. Dr. Jörg Osterrieder

Methods and Algorithms — MSc Data Science

Spring 2026

How Do You Teach a Computer to Decide?

How do you teach a computer to **make good decisions** *without a teacher?*

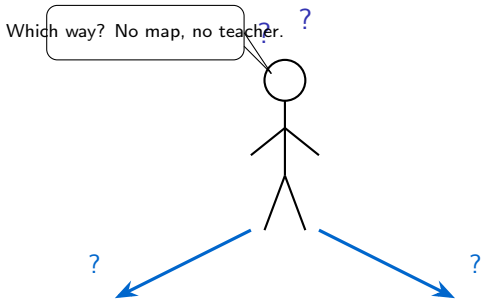
Today: **Reinforcement Learning** — learning by trial and error.

What You Will Learn Today

1. **Describe the RL agent-environment loop**
2. **Explain the explore vs exploit tradeoff**
3. **Identify when RL applies** to real problems

Everything is explained with pictures and analogies.

The Problem: Making Decisions Without Instructions



The agent must learn from experience alone.

RL agents start knowing nothing. They learn entirely from feedback.

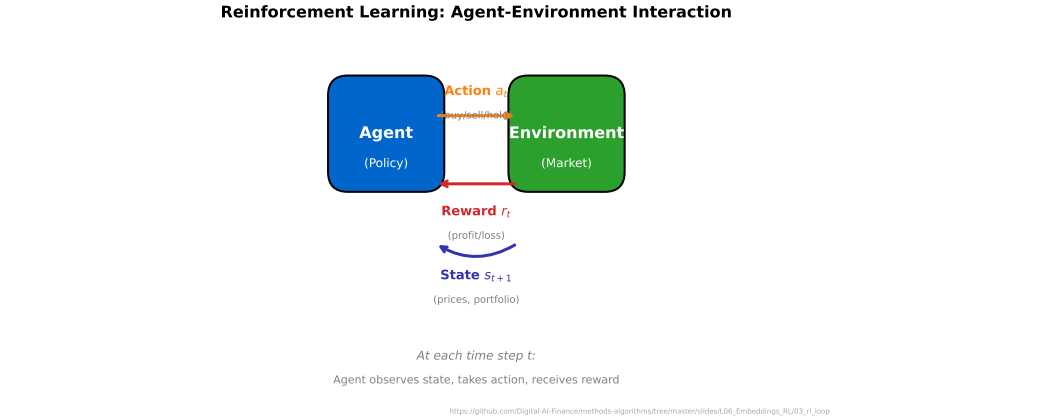
What Makes RL Different?

Paradigm	Signal	Data	Use Case
Supervised	"Here's the answer"	Labeled data	Classification, regression
Unsupervised	"Find patterns"	No labels	Clustering, dim. reduction
RL	**"Try and learn"**	**Rewards/penalties**	**Decision-making, control**

RL is the only paradigm where the agent actively interacts with its environment.

RL is the only paradigm where the agent actively interacts with its environment.

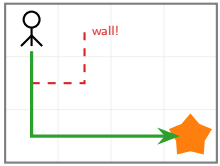
The Core Idea: Act, Observe, Learn, Repeat



- The agent takes an action in the environment
- The environment gives back a reward (or penalty)
- The agent updates its strategy and tries again

This loop is the foundation of ALL reinforcement learning.

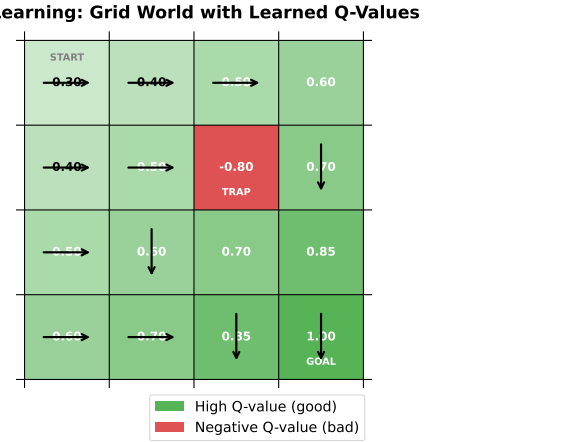
RL: Learning by Trying, Failing, and Trying Again



The agent doesn't know the map — it learns by exploring.

Reinforcement Learning = learn from experience, not from labeled examples.

Watch the Agent Learn the Best Path



https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L06_Embeddings_RL/04_q_learning_grid

- Arrows show the agent's best guess for direction
- After many episodes, arrows converge to the optimal path
- Discovered by trial and error — no teacher told it

Q-learning stores a "quality" score for each state-action pair. Higher Q = better action.

How RL Works (No Math)

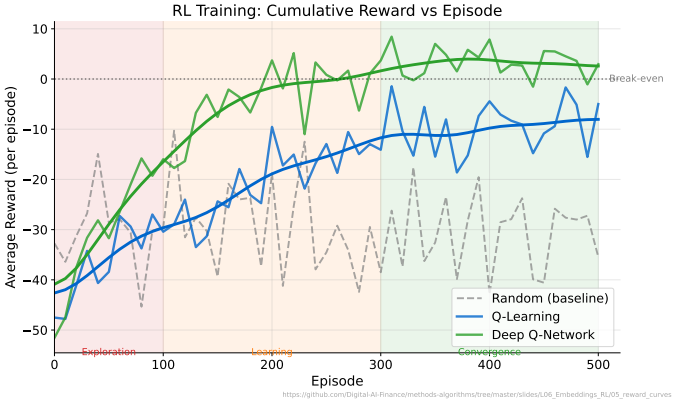
1. **Try an action** — the agent does something (move left, buy, sell) 
2. **Get feedback** — the environment gives a reward ($+1$) or penalty (-1) 
3. **Update strategy** — remember what worked, avoid what didn't 

The Secret

Repeat thousands of times. The agent gets better.

RL agents typically need millions of episodes to learn. That is why simulation is so important.

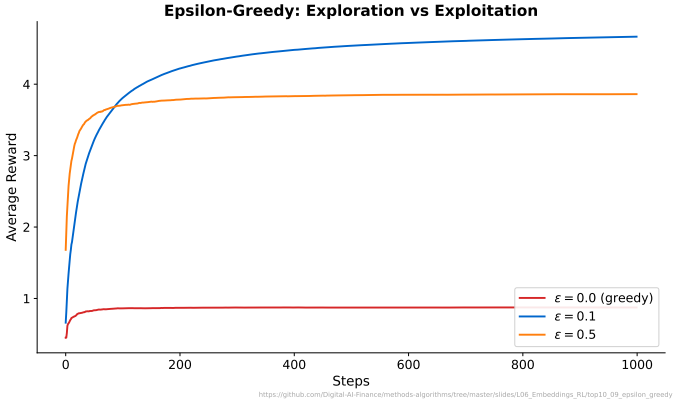
Proof It Works: Rewards Go Up Over Time



- The y-axis shows cumulative reward
- The line goes up = the agent is getting better
- Early episodes are random; later episodes show learned behavior

A rising reward curve is the universal sign of a learning agent.

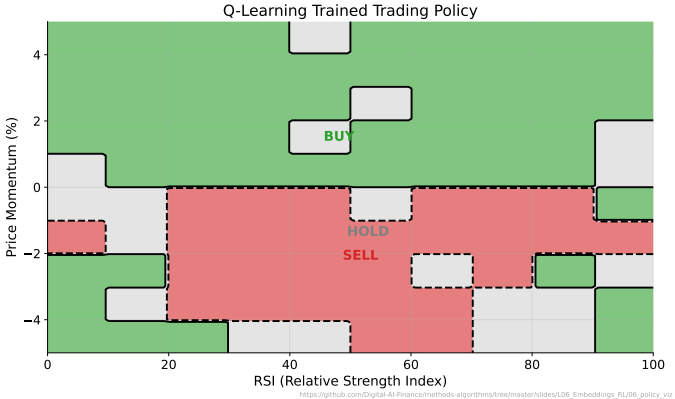
The Key Tradeoff: Explore or Exploit?



- **Explore** = try something new (might find better)
- **Exploit** = stick with what works (safe but might miss better)
- Epsilon-greedy: explore with probability epsilon, exploit otherwise

Too much exploration = never settles. Too much exploitation = misses the best strategy.

The End Goal: A Learned Policy



- A policy maps states to actions
- This is what the agent learned after training
- The goal is an optimal policy — best action for every state

The policy is the agent's strategy — its complete plan for every situation.

Three Things to Watch Out For

1. ! **Needs LOTS of practice** — millions of episodes. That's why we train in simulation first.
2. ! **Reward design is tricky** — the agent finds loopholes you didn't expect. Design rewards carefully.
3. ! **Sim-to-real gap** — what works in simulation may not work in the real world without fine-tuning.

RL is powerful but hungry: it needs massive compute and careful reward engineering.

RL in Finance

- **Algorithmic trading strategies**: learn when to buy, sell, or hold
- **Portfolio rebalancing**: optimize asset allocation over time
- **Optimal order execution**: minimize market impact of large trades

Bottom Line

RL excels wherever you need sequential decisions under uncertainty.

JPMorgan and Two Sigma use RL for trade execution optimization.

Try It Yourself: The RL Recipe

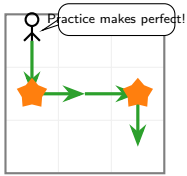
```
for each episode:
  observe state, pick action
  get reward, update strategy
  repeat until done
```

Start simple: OpenAI Gym provides ready-made environments to practice.

---

pip install gymnasium. Try CartPole or FrozenLake as your first RL environment.

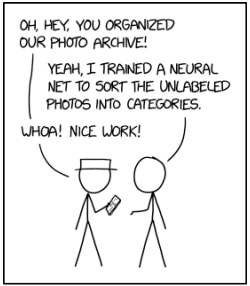
The Expert Agent



1. RL agents learn by trial and error with rewards
2. Explore vs exploit is the key tradeoff
3. Finance uses: trading, portfolio optimization, order execution

RL: learning to make better decisions, one episode at a time.

Go Teach a Machine Something



XKCD #2173 by Randall Munroe (CC BY-NC 2.5). Next: try the Jupyter notebook to train your own RL agent!