Stock Trading with Deep Reinforcement Learning

Sai Vaka, Evan Zimmerman (Group 15)

Introduction & Motivation

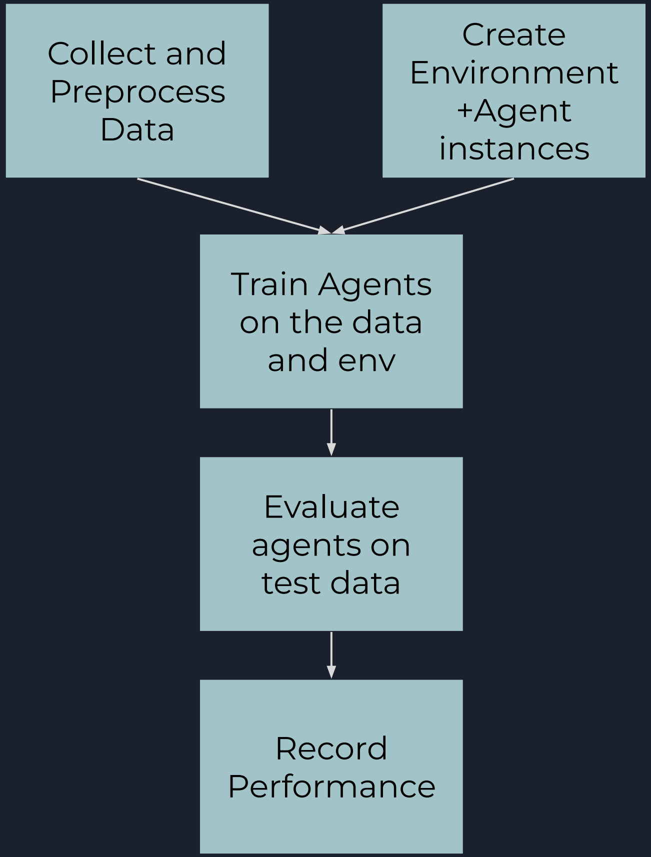
- Traditional trading strategies rely on static models and human decision making
  - This approach has some pitfalls

- Reinforcement Learning is well-suited for developing a strategy to handle complex, dynamic environments (stock market)
  - Removes human emotion

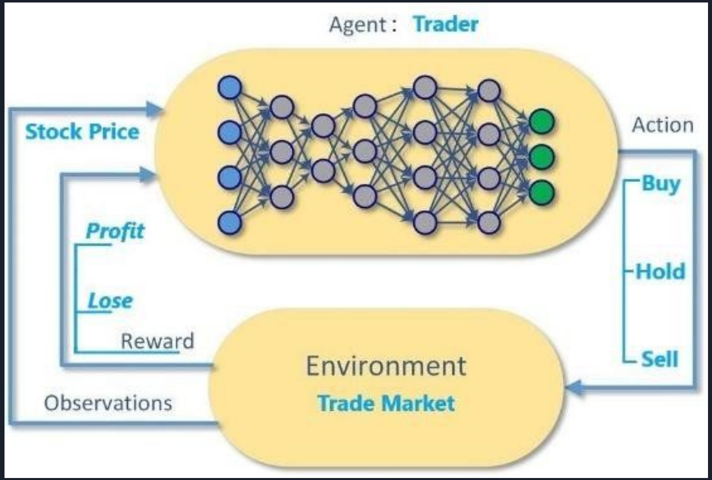
- Think of Stock trading as an MDP, where states are the portfolio/stock prices, actions are buying and selling, and transitions are the changes in prices

Problem Formulation

1. Collect data via yahoo finance api (yfinance, dow 30)
2. Preprocess the data + calculate technical indicators
3. Create our environment (gym)
4. Create our agents (stable_baselines3)
5. Train the agents
6. Test the agents
7. Record results



Environment



Dataset: Dow Jones 30

Observation Space (Daily):

- Shares Held and Price data for all tickers
- Technical indicators for all tickers
- VIX (Volatility Index)
- Balance, Net Worth, Max Net Worth

Action space (continuous):

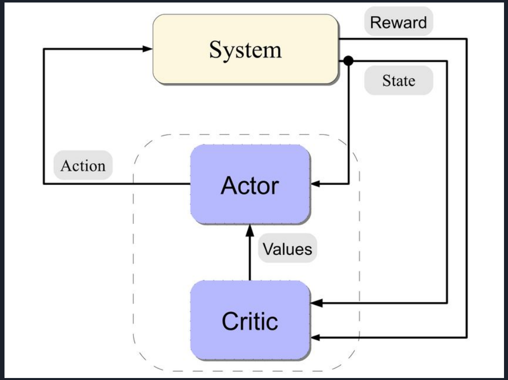
- 30 dim vector where action[i] represents how much to buy and sell
  - action[i] > 0: % of balance to spend on stock[i]
  - Action[i] < 0: % of stock[i] to sell
  - Action[i] == 0: hold

Reward:

- Change in net worth
  - Subtract shared held when VIX > 30

Actor - Critic Method for RL

1. **Actor**:
    - Represents the **policy** that decides what actions to take given a state.
    - Updates the policy directly by learning how to act in the environment to maximize the expected return.
    - Output: action(s)
2. **Critic**:
    - Estimates the **value function**, which evaluates the quality of the actions taken by the actor.
    - Helps the actor improve by providing feedback in the form of value estimates
        1. state-value V(s),
        2. action-value Q(s,a),
        3. advantage Q(s,a) - V(s)

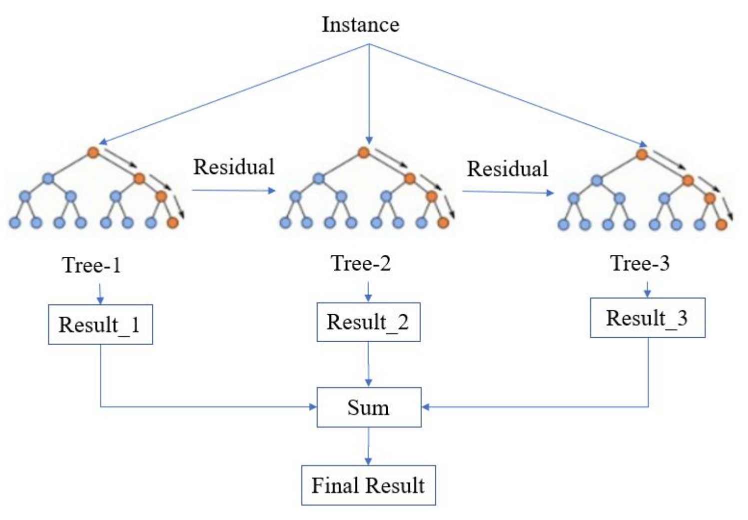


RL Agents

Trained from 2009-01-01 to 2016-12-31 (10,000 total training steps)

Algorithm	On/Off-Policy	Action Space	Key Feature	Stability
PPO (Proximal Policy Optimization)	On-policy	Both	Clipped policy updates	High
A2C (Advantage Actor-Critic)	On-policy	Both	Advantage estimation	Moderate
DDPG (Deep Deterministic Policy Gradient)	Off-policy	Continuous	Deterministic policy, experience replay	Low
SAC (Soft Actor-Critic)	Off-policy	Continuous	Entropy-based exploration	High
TD3 (Twin Delated DDPG)	Off-policy	Continuous	Twin Q-networks, delayed updates	High

Ensemble Agents



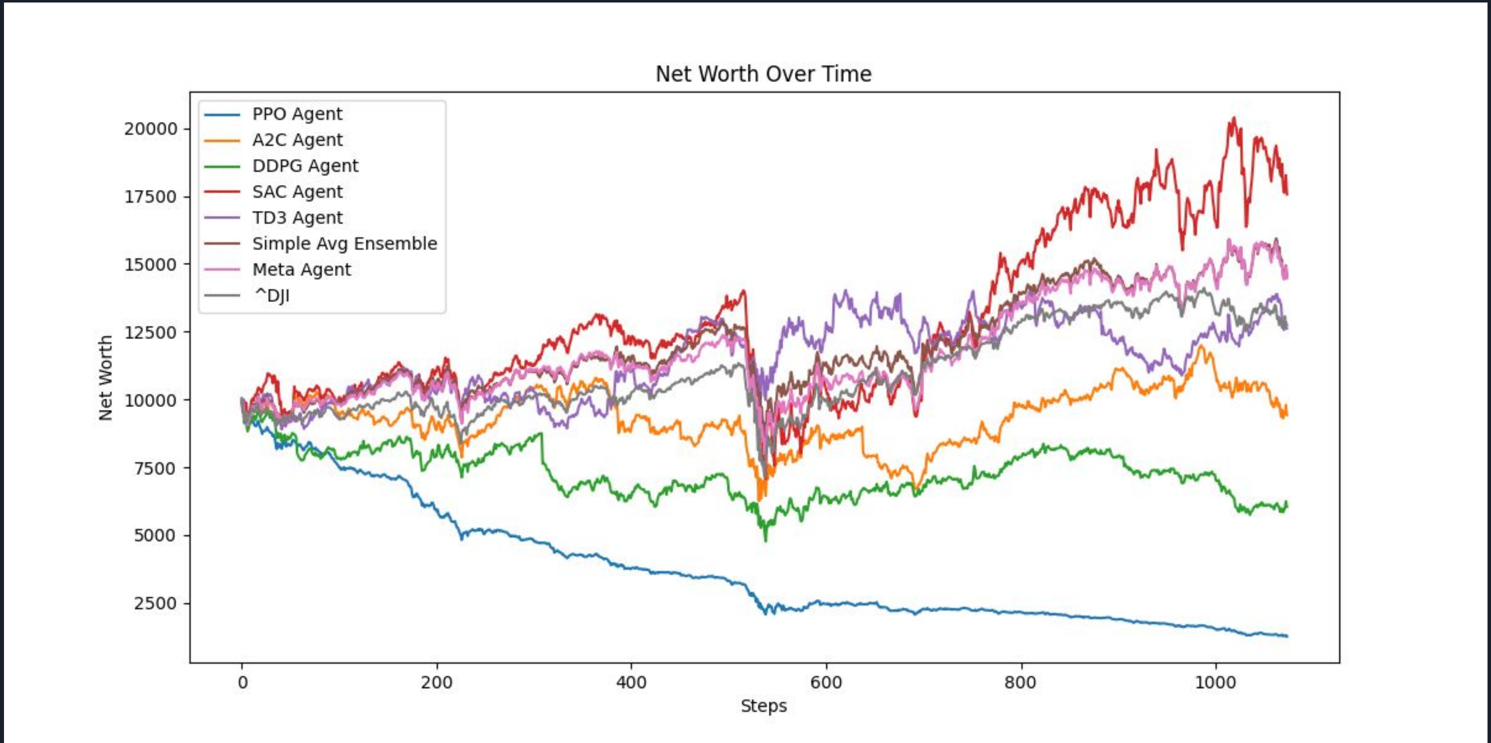
Simple Average

- Take the average of all agent’s actions

Meta Agent

- Takes as input all agents actions
- XGBoost model to predict the best action
- Trained on the best actions over a training period

Results (01/30/2018 - 05/08/2022)



Results Table

	PPO	A2C	DDPG	SAC	TD3	Simple Avg	Meta Agent	DJI
Cumulative Return	-87.35%	-5.32 %	-39.3%	**76.44%**	28.04%	46.17%	52.54%	25.63%
Annual Volatility	0.2349	0.2854	0.2664	0.3711	0.2647	0.2557	0.2932	0.2218
Max Drawdown	-87.35%	-42.09%	-52.14%	-49.57%	-24.90%	-34.63%	-36.47%	-37.08%

Q&A