Optimized Kalman Filter

Mathys Vinatier

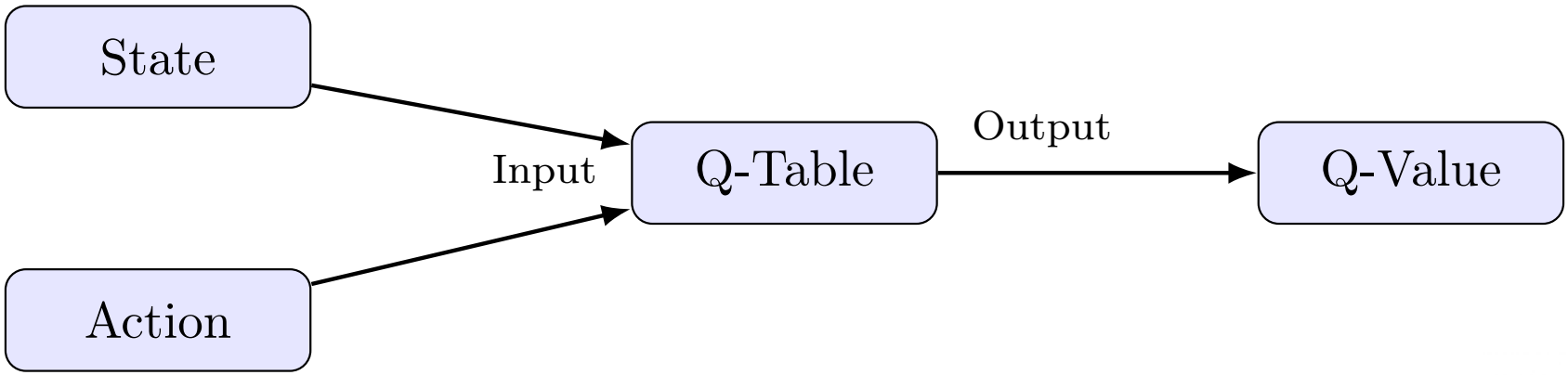
Seoul National University

August 13, 2025

Epsilon-Greedy Policy

- **Q-table:** Internally, Q-Learning uses a Q-table storing values for each state-action pair. Initially, all values are zero, and the table is updated iteratively during training.

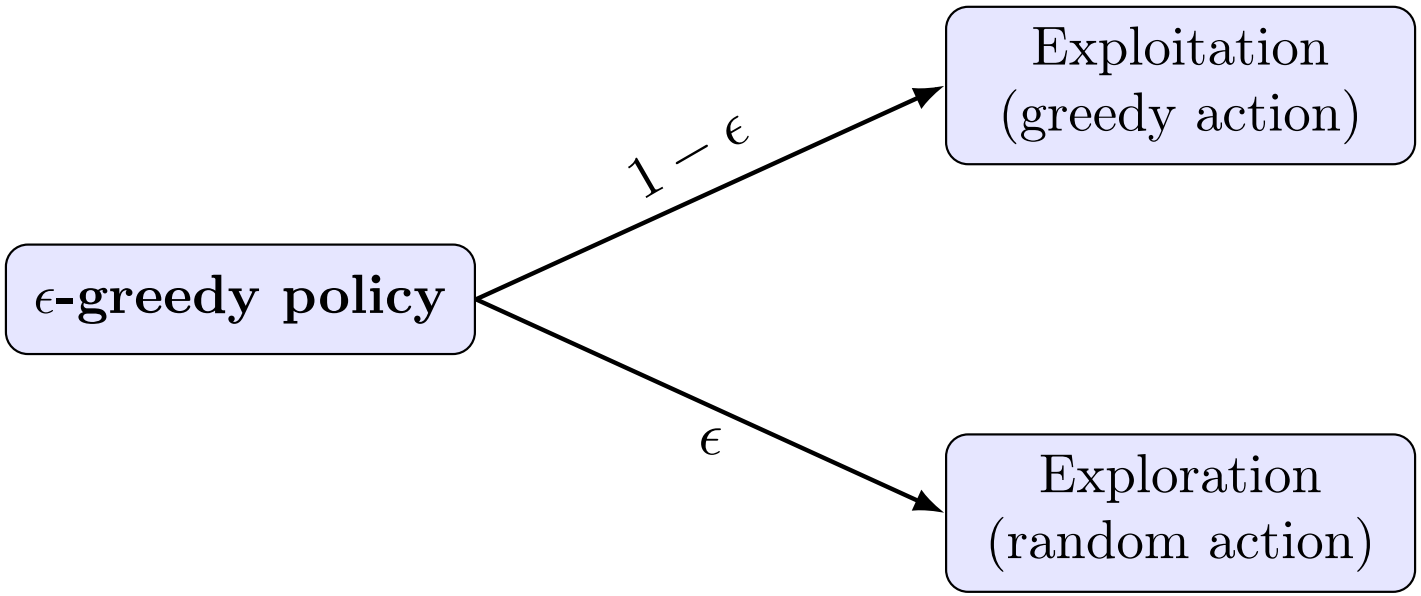
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$



Epsilon-Greedy Policy

- **Epsilon-Greedy Strategy :** To balance exploration and exploitation, actions are chosen using an epsilon-greedy policy

$$\pi^*(s) = \arg\max_a Q^*(s, a)$$



Environement Implementation

TradingEnv

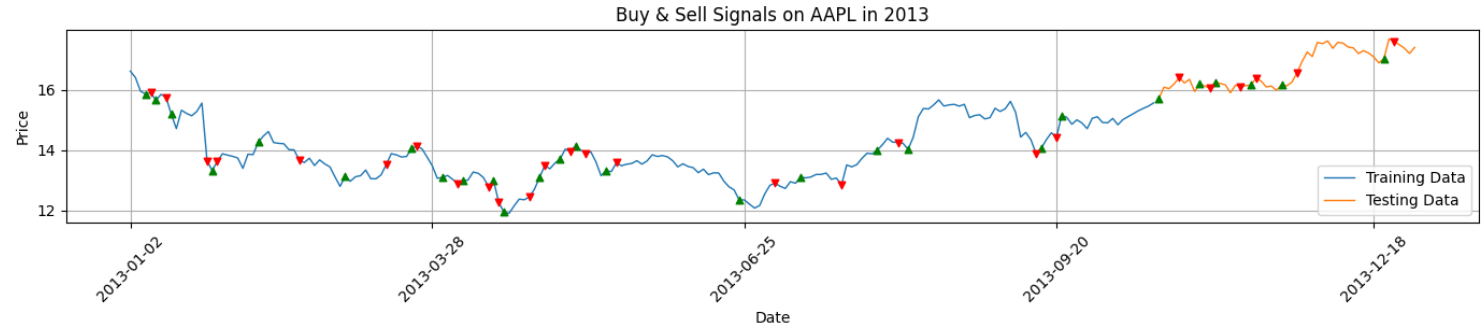
- df : DataFrame
- n_steps : int
- current_step : int
- initial_balance : float
- balance : float
- position : int
- last_action : int
- observation_space : Box
- action_space : Discrete

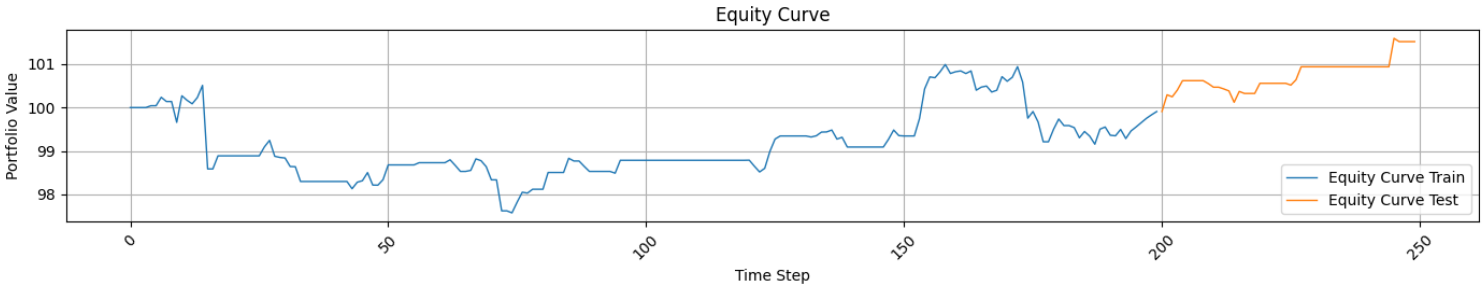
+ __init__(df)
+ _get_obs()
+ sample_valid_action()
+ get_valid_actions()
+ step(action)
+ set_data(df)
+ reset()

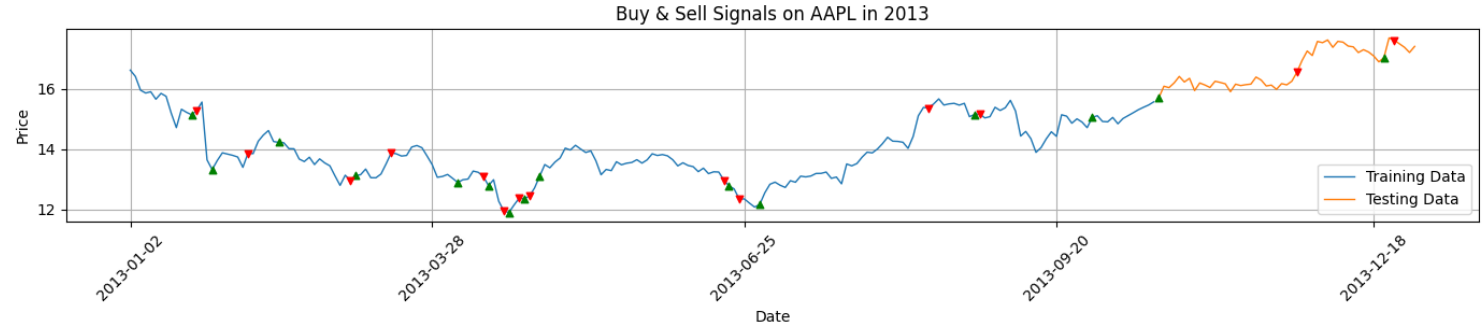
Algorithm 1: Trading Environment Behavior

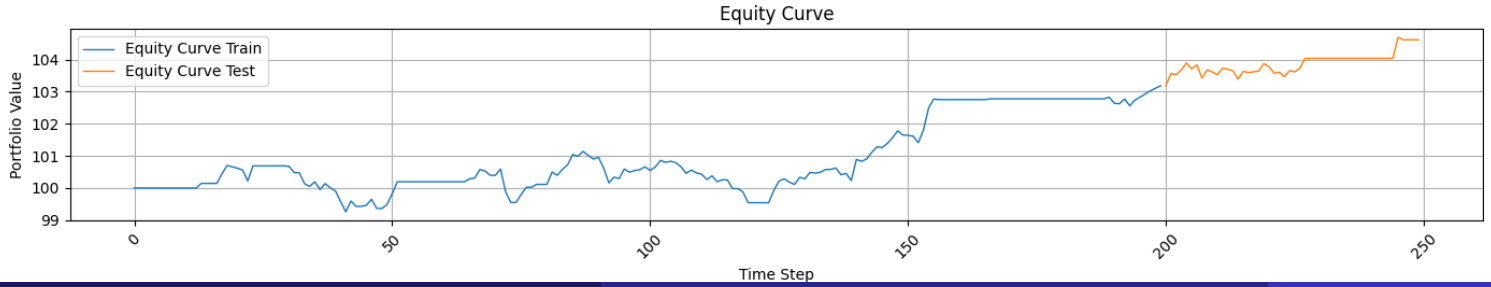
Input: Environment env, number of episodes N, exploration rate ϵ
Output: Episode value and rewards
for $episode \leftarrow 1$ **to** N
  $s \leftarrow env.reset()$;
  total_reward $\leftarrow 0$;
  while *True*
    Get valid actions $A_{valid} \leftarrow env.get\_valid\_actions()$;
    Select $a \leftarrow$ random choice from A_{valid};
    Execute $(s', r, done, info) \leftarrow env.step(a)$;
    total_reward $\leftarrow$ total_reward $+r$;
    Log $(episode, s, a, r, info["portfolio\_value"])$;
    $s \leftarrow s'$;
    if *done* **then**
      break
  Print episode summary : total_reward, final portfolio value;

Results

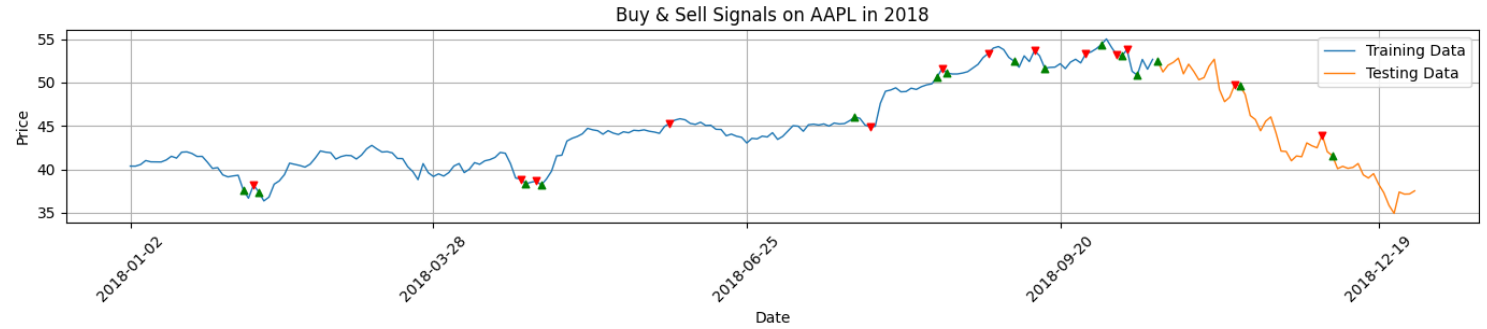

Buy & Sell Signals on AAPL in 2013
Price
Date
Training Data
Testing Data
2013-01-02
2013-03-28
2013-06-25
2013-09-20
2013-12-18

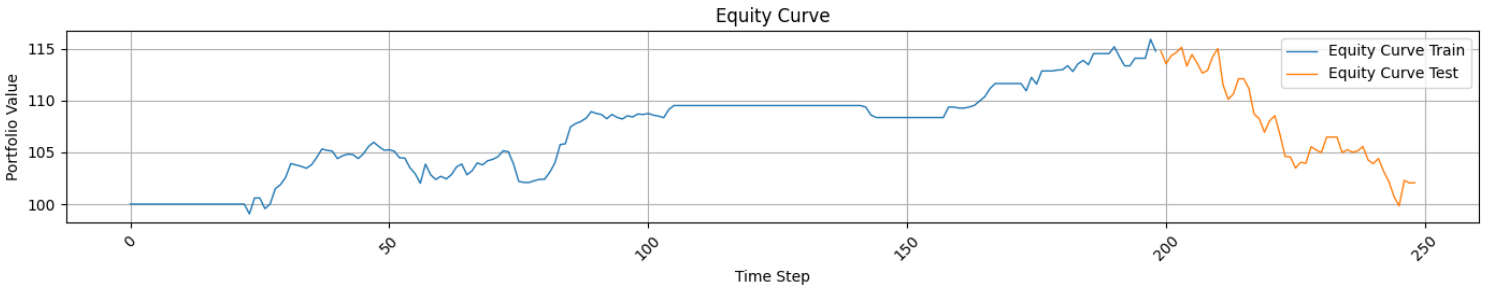

Equity Curve
Portfolio Value
Time Step
Equity Curve Train
Equity Curve Test

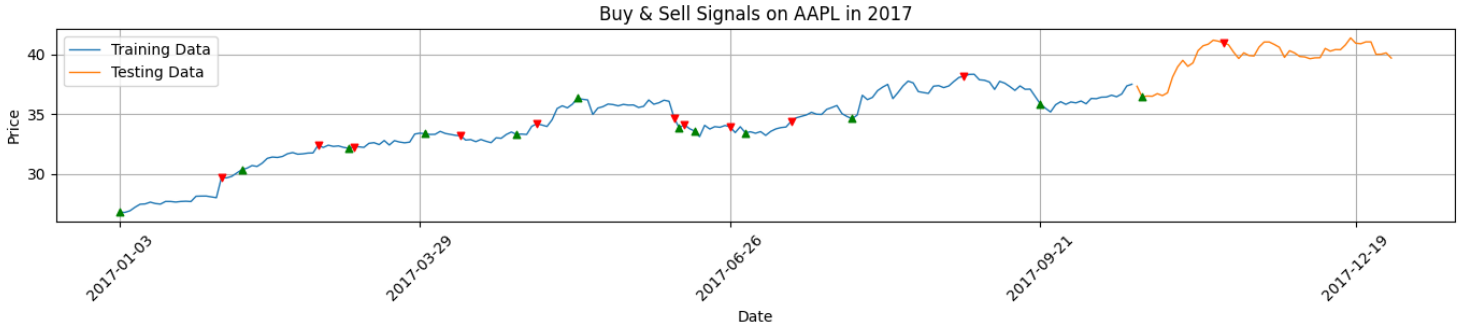

Buy & Sell Signals on AAPL in 2013
Price
Date
Training Data
Testing Data
2013-01-02
2013-03-28
2013-06-25
2013-09-20
2013-12-18

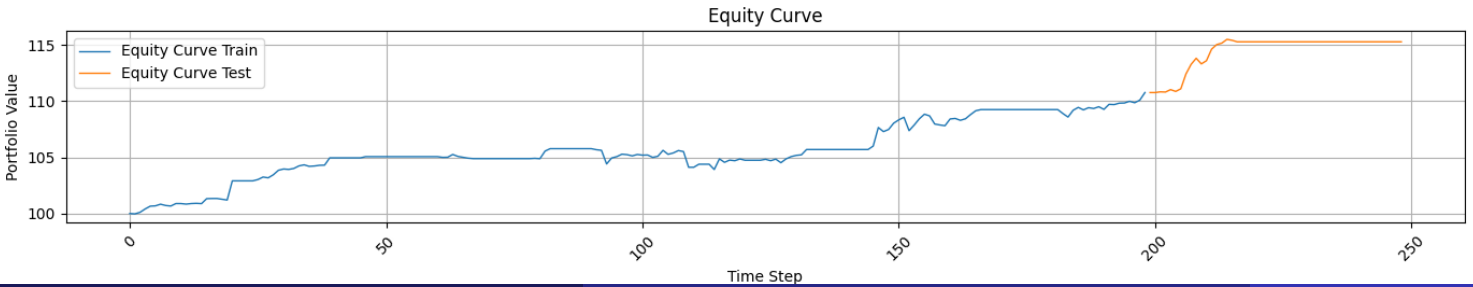

Equity Curve
Portfolio Value
Time Step
Equity Curve Train
Equity Curve Test

Results issues









Preparing PPO

- Designing and integrating a deep neural network architecture
- Compare DQN and Q-Learning greedy policy
- Expanding to more realistic market conditions
- Hyperparameterization

Thank You