Project RL - Interim Presentation

Clàudia Domènech
Alexia Spinei
Satiga Godrie

The Environment

Time step: 1 hour

- One environment step = one hourly spot price
- Episode ends at the end of the price time series

States

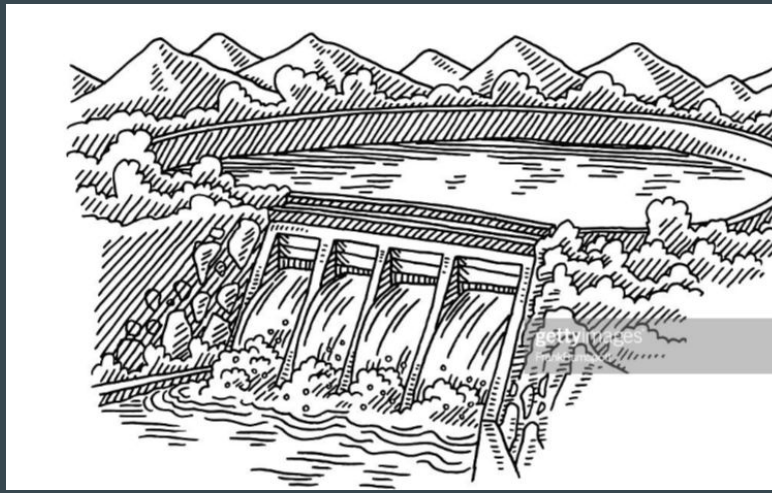
- Reservoir level W_t / W_{max}
- Current electricity price P_t

Actions: Idle, Sell, Buy

Physical constraints: **max flow** and **storage capacity**

Reward (profit per hour)

- Sell = $+ P_t \cdot (0.9 \cdot E_{pot})$
- Buy = $- P_t \cdot (E_{pot} / 0.8)$
- Idle: 0



Baseline Model: Rule-Based Threshold Policy

Baseline idea:

- Store energy when prices are low
- Sell energy when prices are high
- Idle otherwise

Decision rule

- If P_t is in the lowest 33% of prices → Buy
- If P_t is in the highest 33% of prices → Sell
- Otherwise → Idle

Thresholds:

Buy price (P_buy): 33rd percentile of training prices
Sell price (P_sell): 67th percentile of training prices

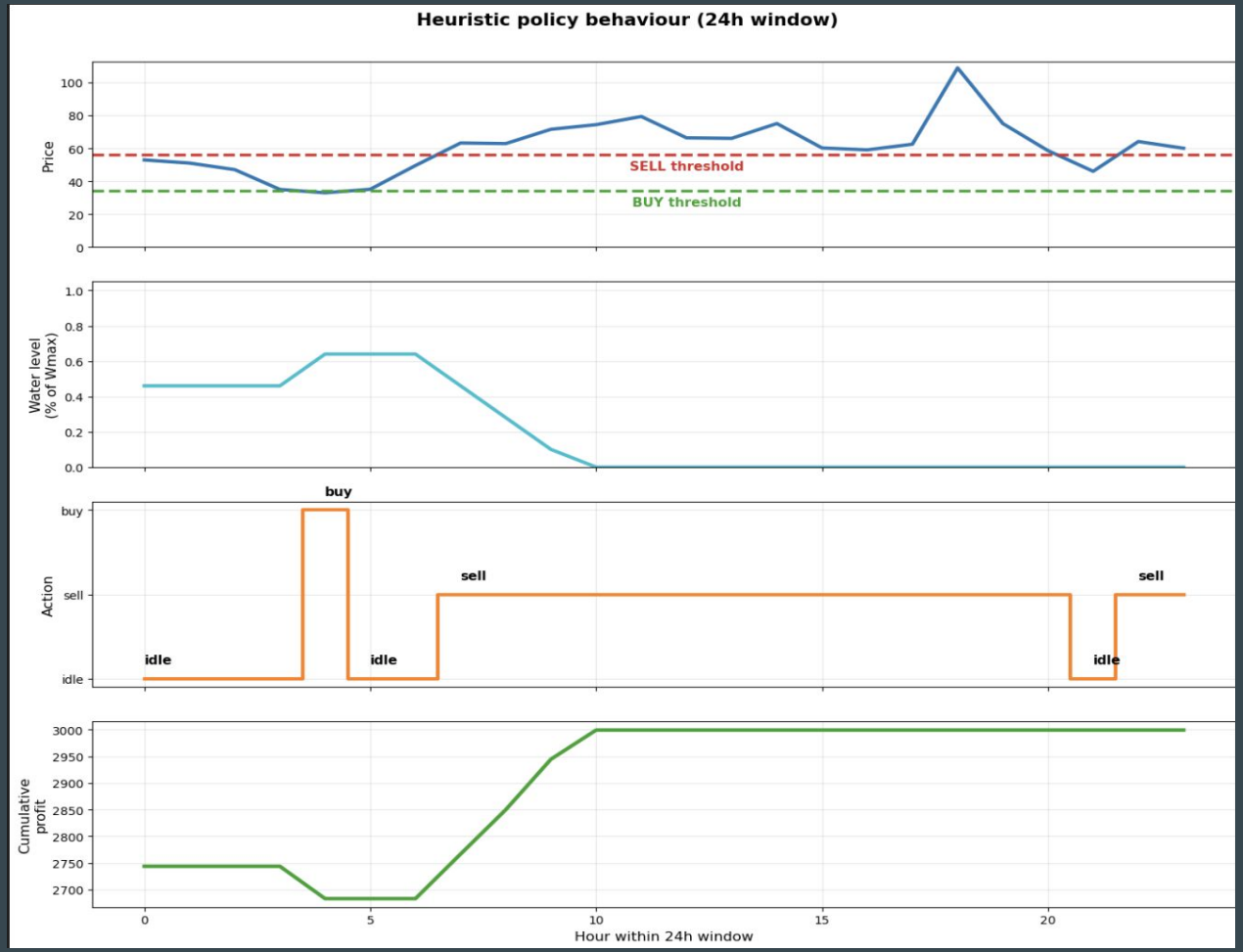
Computed once from training data and fixed during validation.

Visualizing Results (1min 30 sec)

- plot averaging timestamps (cumulative reward)
- plot 24h - clear example heuristic
  - cumulative reward
  - action - sell
  - price
  - water level
- optional: rnd baseline

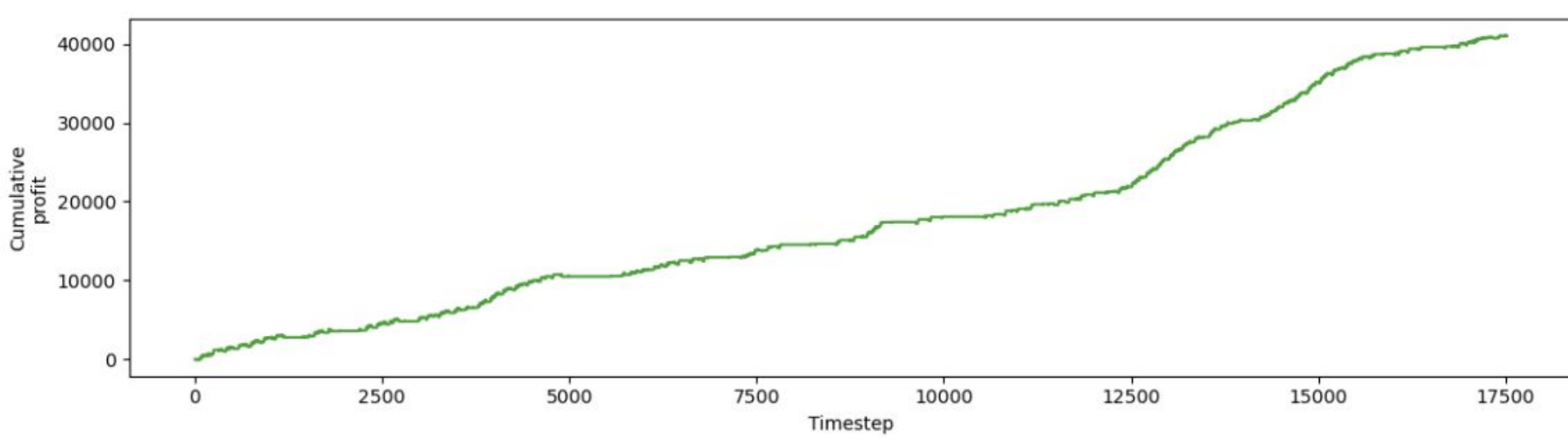
Visualizing Results

24h window:



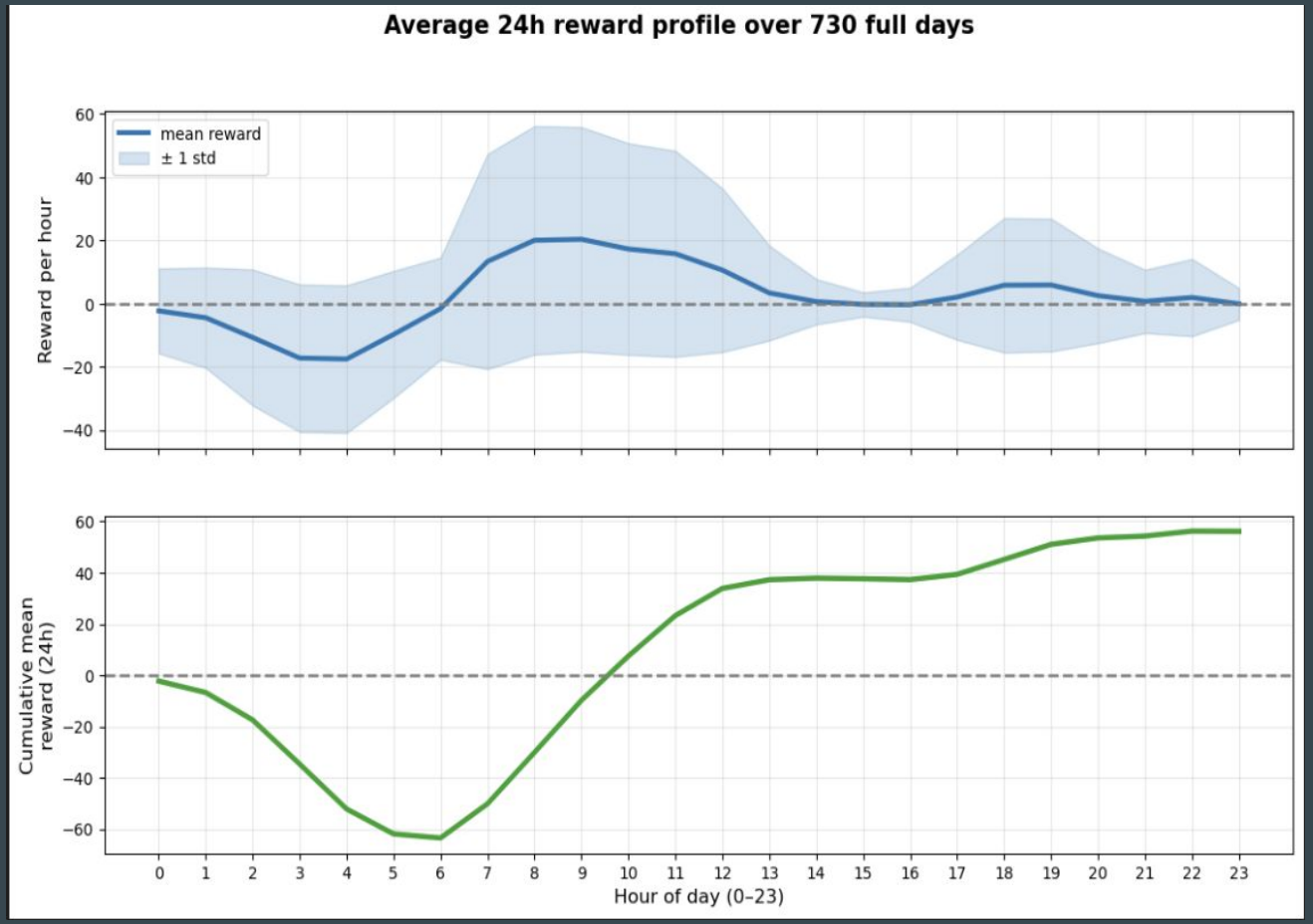
Visualizing Results

Cumulative reward over total timestamps:



Visualizing Results

Averaging timestamps:



Future Steps Implementation

From Baseline to RL

- Tabular RL: discretize storage and price, learn a Q-table
- Validation: compare learned policy to the baseline
- Maybe Deep RL: move to continuous states and richer features

Future Steps

1. Feature engineering

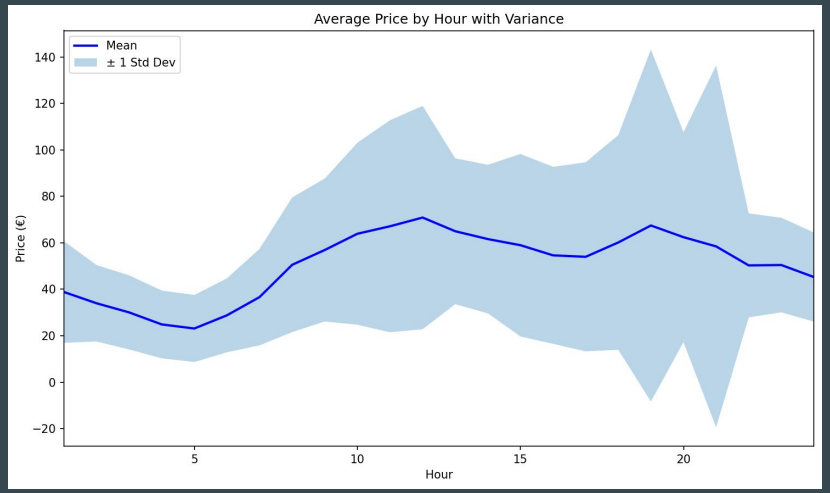
2. Reward shaping

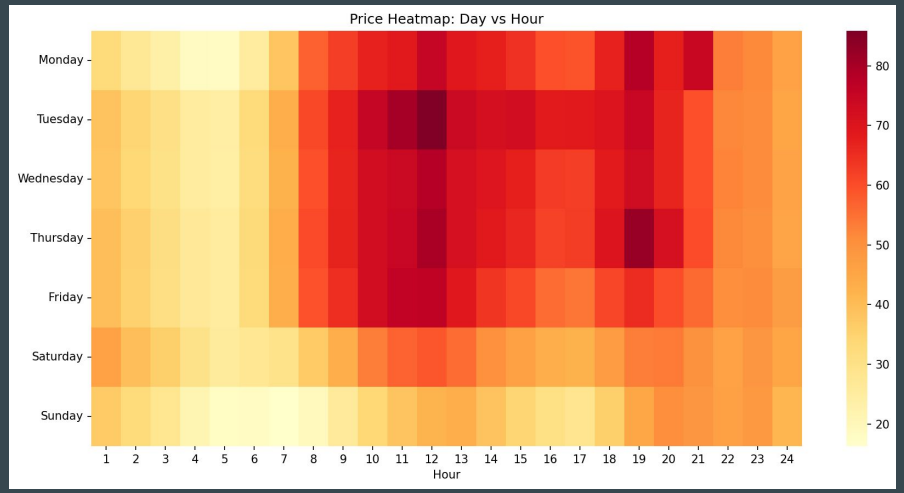
State = (storage_level, price, , , ,)

Feature engineering

Reward Shaping

Grouping hours into day segments



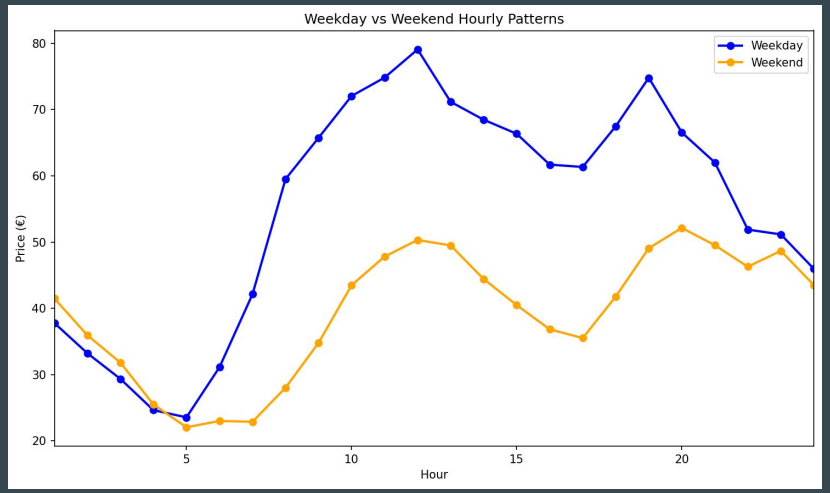


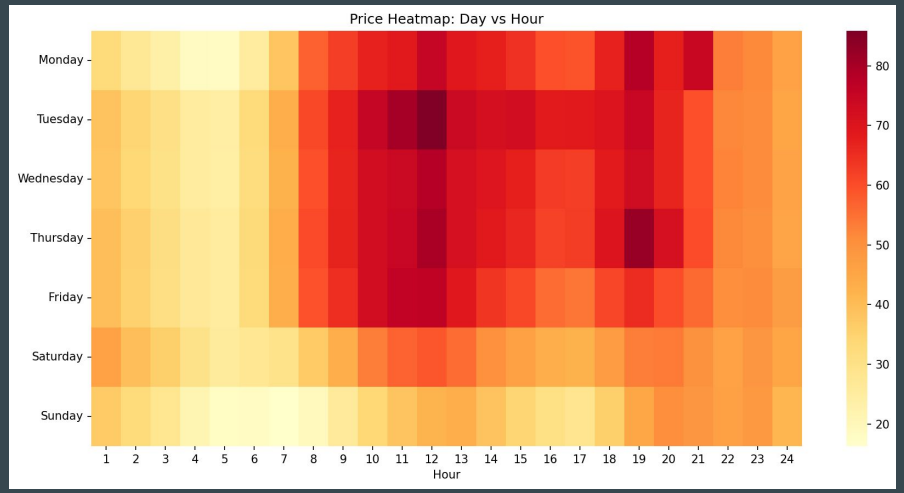
State = (storage_level, price, hour_period, , ,)

Feature engineering

Reward Shaping

Distinguishing between weekdays and weekends

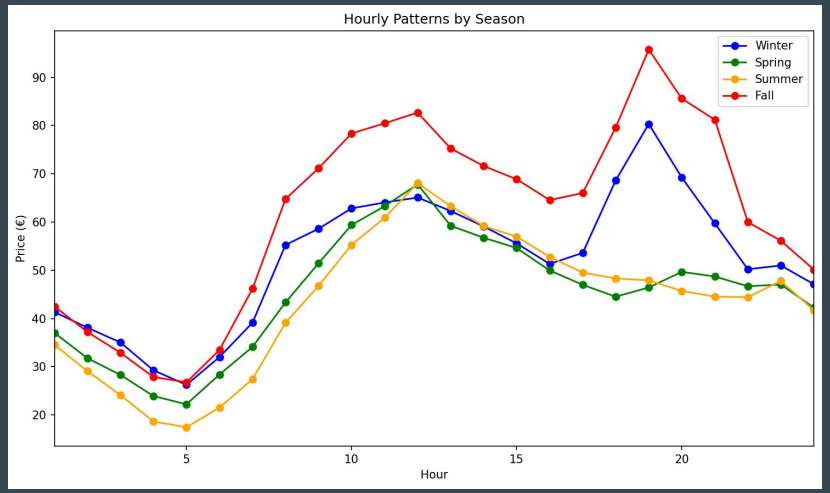




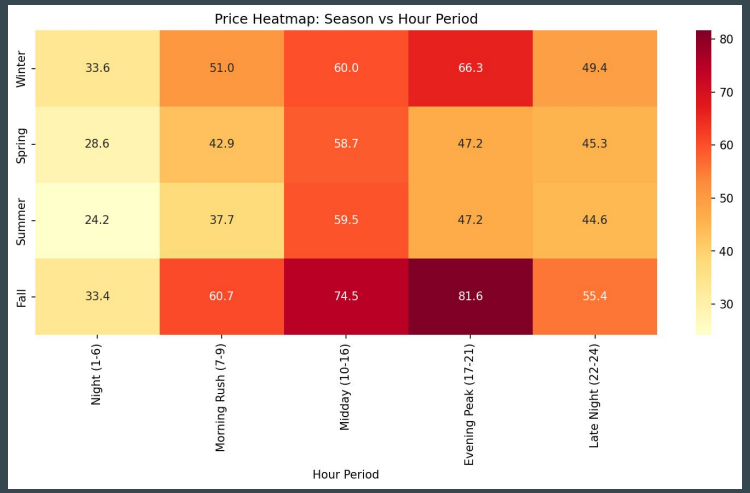
State = (storage_level, price, hour_period, is_weekend, ,)

Feature engineering

Grouping months into seasons



Reward Shaping

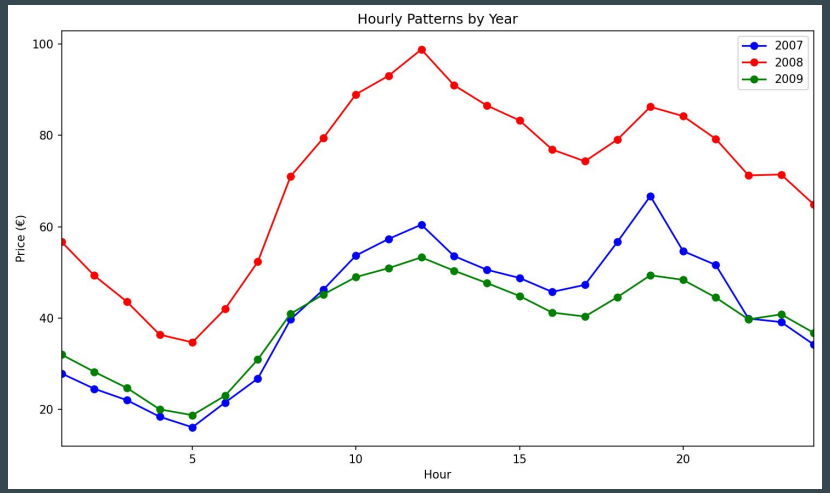


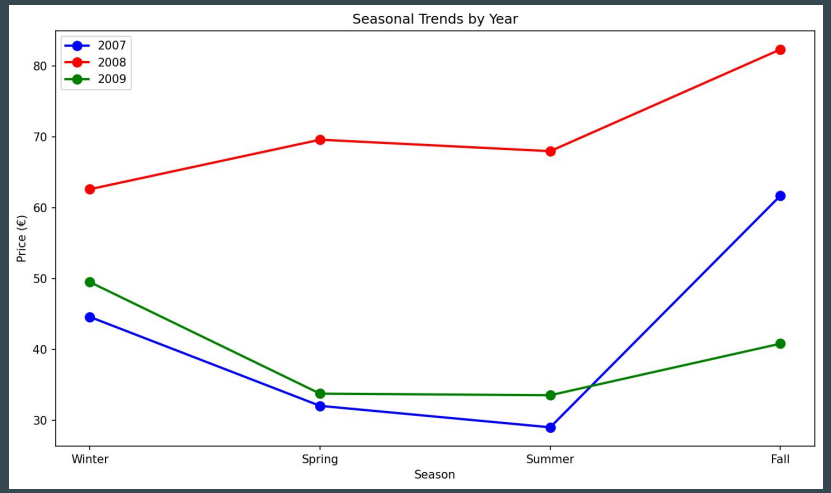
State = (storage_level, price, hour_period, is_weekend, season,)

Feature engineering

Reward Shaping

Price normalization addressing interannual variability



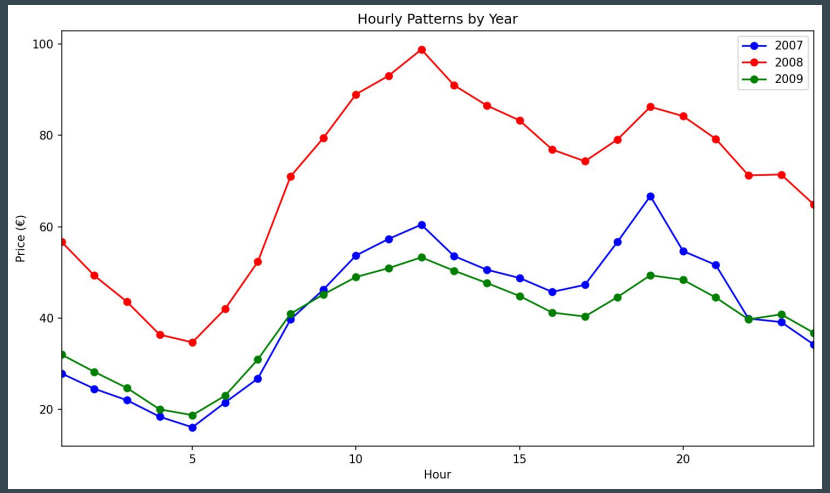


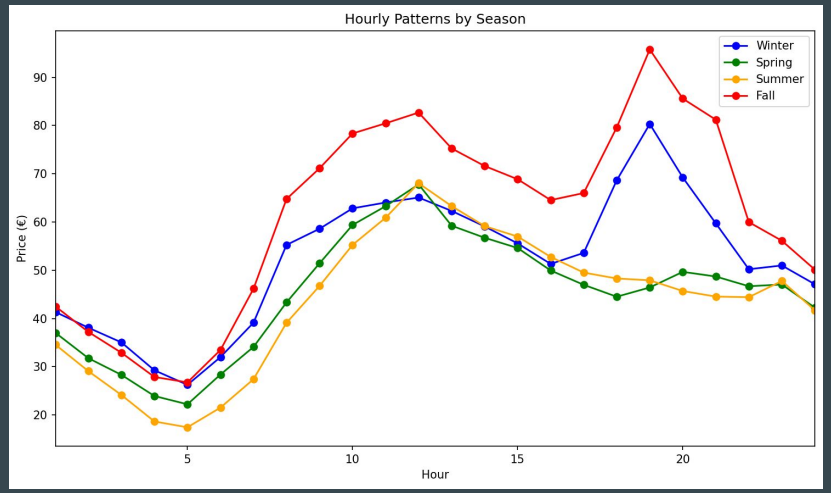
State = (storage_level, price, hour_period, is_weekend, season, normalized_price)

Feature engineering

Reward Shaping

Penalizing empty storage when entering peak hours





State = (storage_level, price, hour_period, is_weekend, season, normalized_price)

Future Steps Implementation

State = (storage_level, price_bin, hour_period, is_weekend, season, normalized_price)

Reward Shaping : Penalize low storage entering peak hours

Thank you for listening

...

Clàudia Domènech
Alexia Spinei
Satiga Godrie