**Environment:**

This is a single stock environment. It means that at each episode one single stock is **randomly** selected from a pull of stock data. I have selected 25 stocks from Indian market which have high volume and also high beta.

Past 3 months historical data is used for training. And next (from test data time) 1 week data is used for testing.

**1. Observation:**

The main goal is to train an agent to identify actual or legitimate momentum in the stock and avoid false momentum. This is why it is essential to include observation parameters humans use for momentum trading. Also, the parameters need to be normalized so that it works for all the stocks.

Parameters:

1. NIFTY Parameters
2. Sector-Related Parameters
3. Close Price: Instead of the closing price, I am using ROC, Relative Price to Moving Average (PMA)

**1. ROC\_5, 2. ROC\_21, 3. ROC\_55**

**4. PMA\_5, 5. PMA\_21, 6. PMA\_55**

1. Volume: Rate of Change (ROC), Relative Volume (RVOL)

**7. ROC\_5, 8. ROC\_21, 9. ROC\_55**

**10. RVOL \_5, 11. RVOL \_21, 12. RVOL \_55**

1. RSI

**13. RSI\_5, 14. RSI \_21, 15. RSI \_55**

1. Relative Strength NIFTY

**16. RS\_5, 17. RS \_21, 18. RS \_55**

1. Money Reserve: **19.** **Relative to the Initial Fund**
2. Stock Holdings: **20.** **In Money Relative to the Initial Fund**
3. **21. Action Taken at Previous Step**
4. **22. Reward at the Previous Step**

**2. Action:**

This is a long-only environment. No shorting action is allowed.

Action space is discrete +ve integers. It is scalled in following way internally e.g., [0 to 11] is scalled [-5 to +5].

Position\_per\_action\_value = percentage of fund will be used to buy or sell per unit int value in action.

Example: For an action space [10 to -10]. If action value is 2, 20% of the cash balance will be used to buy stock. Also, if action value is -4, 40% of the stock holding will be sold.

**3. The trade execution mechanism (Indian Market):**

**3.1. Random selection of a stock:** At every episode, a random stock is selected from the list of stocks, i.e., list\_of\_stock.

**3.2. Stock data:** The environment needs historic market data (.csv files) in OHLC format which is saved in the data directory, i.e., data\_dir.

**3.2.1.** The index column must be in datetime format and named as ‘date’.

**3.2.2.** The input dataframe must have ‘open’, ‘high’, ‘low’, ‘close’, and ‘volume’ columns

**3.3. Random time duration:** At every episode, a random time duration of the episode length from the data frame is selected. The length of the episode (episode\_length\_in\_steps) is user-defined.

**3.4. Trading Fee:** At every step, a trading fee is needed to pay. The total fee is ((trade\_volume\_in\_money × fee\_percentage)/100) or max\_fee, whichever is the minimum.

**3.5. Execution Price:** The execution price is a random price between the high and low price of the time step.

**3.6. Buy and Sell mechanism:** In India, fractional stock buy and sell is not allowed.

**3.6.1.** Buy action execution: Buy action is executed when action\_value > 0

The target position is given by,

target\_position\_in\_money = self.money\_reserve × (1 – (self.max\_fee /100)) × (action\_value/100)

The term, (1 – (self.max\_fee /100)) ensures that there is enough money left to pay the fee. When the fee is much smaller this approximation should not affect the environment significantly.

The actual stock buy quantity is

stock\_buy\_in\_quantity = int(target\_position\_in\_money / execution\_price)

**3.6.2.** Sell action execution: Sell action is executed when action\_value < 0

The stock sell quantity is given by

stock\_sell\_in\_quantity = int(self.stock\_holdings\_in\_num × (-self.action/100))

**3.7. Episode Ending:** An episode ends is the maximum number of steps, i.e., (episode\_length\_in\_steps) is over of the net\_worth is reduced more than the acceptable loss, (max\_acceptable\_drop\_down\_percentage\_episode\_stopping).

**4. Reward:**

Reward is function is designed as follows

Reward(t) = ln(net\_worth(t)/self.old\_net\_worth(t-1))

I have adapted it from Gym Trading Environment [1]

[1] https://gym-trading-env.readthedocs.io/en/latest/environment\_desc.html#reward

Seems to be legit because

So, if all the reward of all the time steps of an episode are summed it will be

The summation will be equal to

However, I have to explore more for finding better Reward Function.

Seems to be legit. However, I have to explore more.

Cause ln(a/b) = ln(a) + ln(1/b)

**5. Negative Reward for Wrong Actions:**

**Buy Action: If money reserve < one stock price**

**Sell Action: If stock reserve = 0**

**Process: make a variable**