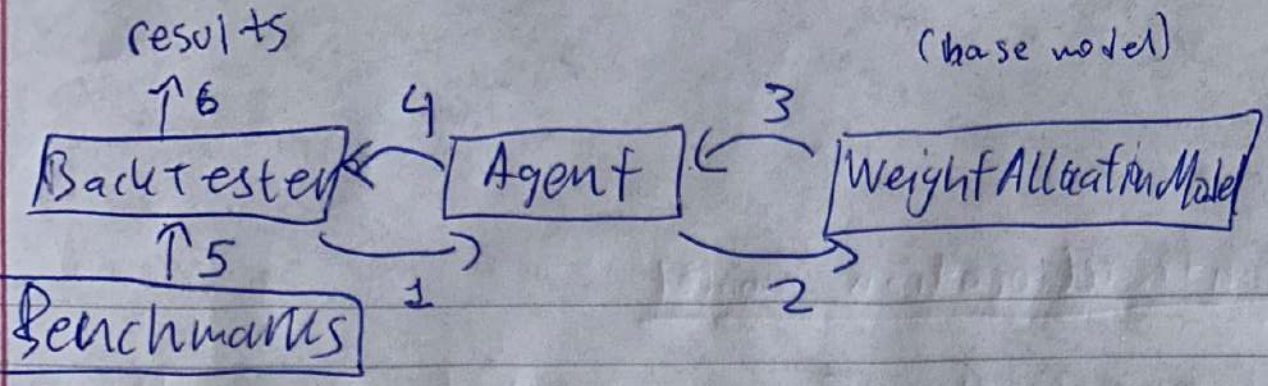

Backtester

Backtester generic flow as follows:

1) self.get_data()
2) self.agents_allocate()
3) results = self.evaluate_agents(self.benchmarks)

} def run

} def run_n_evaluate

Agent

- Based on the model that is passed as an argument in the agent, the aperopriate weights_allocate method is called
- date_data_needed → function to have the min(data_from) that the model requires (e.g. for 3 months back we would need 3 months back outside of our backtesting period.

Weights_allocation_model

- Has essentially 2 main methods, the same as the agent
- weights_allocate inputs: (date_from, date_to, tickers, data)
  dt.date, dt.date, list, pd.DF
- weights_allocate outputs:

  It outputs the dataframe with the updated weights for each time that we have updated them.

Evaluation Classes

- Base class, that outlays some structure, on how the evaluation specific classes are created.
- The benchmarks are selected for a type ("phl", sharpe etc) and a frequency ("weekly", "period", "monthly").

Backtester output

- results {} → nested dict that contains the benchmark results for each agent.
- The results are displayed into console, and saved to excel for the given simulation.