1 Funcs.py

`def multi_hypo_test(data, hypo, prior):`

Perform hypothesis testing on binary data. Binary data means the observed data can only take two values, like good or bad for a product. Given the observed data, calculate the posterior probabilities for different hypotheses.

Parameters:
- data: A list of binary values, e.g., [0, 1, 0, 1, 0, 1, 0, 1, 0, 1].
- hypo: A list of probabilities for each hypothesis being true, e.g., [0.5, 0.5] represents the probability of the data being 0 under each hypothesis.
- prior: A list of prior probabilities for each hypothesis, e.g., [0.5, 0.5].

Returns:
- posterior: A list of posterior probabilities for each hypothesis.
- dB: The calculated decibel values based on the odds of the hypotheses.

Reference:
This method is based on the work in Jaynes' "Probability Theory: The Logic of Science", Chapter 4.

`def best_strategy(odds):`

Assume the only types of wagers allowed are to choose one of the outcomes i, i=1,...,m, and bet that i is the outcome of the experiment. Use the arbitrage theorem to determine if there is a sure-win betting strategy. If such a strategy exists, solve it using linear programming through the simplex method.

Parameters:
- odds: A array of odds for each outcome.

Returns:
If a sure-win strategy exists, the function returns the optimal values for each bet and the value of v (profit margin). Otherwise, it prints that no sure-win strategy exists.

Reference: This method is based on "Introduction to Probability Models, 11E" by Sheldon M. Ross, Chapter 10. See the example 10.2 for more details.

`def markov_stable_state(x, p, method='past_coupling', x0=None, alpha=None):`

Generate the state of a stationary Markov chain using two methods: past coupling or an alternative method.

Parameters:
- x: A list of possible states, e.g., [1, 2, 3, 4].
- p: A transition matrix where `p[i][j]` indicates the probability of moving from state `i` to state `j`. For example:
[[0.1, 0.2, 0.3, 0.4],
[0.2, 0.3, 0.4, 0.1],
[0.3, 0.4, 0.1, 0.2],
[0.4, 0.1, 0.2, 0.3]].
- method: `'past_coupling'` to use the past coupling method or `'other'` for an alternative method (default is `'past_coupling'`).
- x0: The initial state (required if using the alternative method).
- alpha: The event occurrence rate (required if using the alternative method).

Returns:
The steady state based on the chosen method.

Reference: This method is based on "Introduction to Probability Models, 11E" by Sheldon M. Ross, Chapter 11.

2 InterestingCases.ipynb

1. `False positives`
2. `Example 2.53 from Ross' book`
3. `A efficient sorting algorithm from Ross' book`
4. `The probability of winning a Badminton game based on the probability of each point being won`