CSE 516: Homework 4

Due: April 28, 2015 by 5 PM outside Jolley 510

Note: There are two questions. You may work on Problem 2 in a team of two or three people (and submit only one writeup). For Problem 1, please do your work individually. The quality of your writeup will be a **huge** component of your grade on this homework.

1 Prediction markets (30 points)

From Thursday, April 16 at 1:05 PM, to Thursday, April 23rd at 11:00 AM (all times Central Daylight), you will have the opportunity to participate in two prediction markets which can be accessed at `http://simla.cse.wustl.edu/` (your accounts should all be reset to the same state as each other before trading begins: until that point feel free to continue exploring the test market we have set up).

1. Shares in the first market will liquidate at 100 if the local temperate at 11:42 AM on April 23rd is at or above 62 degrees Fahrenheit, as indicated on the "Dark Sky" app on Sanmay's phone, and 0 otherwise.

2. Shares in the second market will liquidate at 100 if there are at least twenty students who are enrolled in this class present in class at exactly 11:38 AM on April 23rd, and 0 otherwise.

You have been given an endowment of MASecoins to trade with, and your total earnings will count towards your overall MASecoin score. You may trade as you wish during the game, but for the purposes of this homework problem, write down what you would consider good trading strategies for these markets, and what are the important features that make them different from each other. Things you should definitely think about in both cases include (1) how to estimate the probability of the outcome that each share pays off 100, especially as a function of how much time is left in the market and your current knowledge; (2) how to trade given your current holdings, the market price, your budget, your current estimate of the probability, and how much time is left in the market; (3) how the different ways in which the market outcome is determined in the two different markets should change your trading strategy.

Your homework grade will only depend on your answer to this question, not your trading performance.

2 A trading bot (70 points)

You will write a trading strategy for prediction markets mediated by the logarithmic market scoring rule (LMSR) market maker described in class. Your trading algorithm has access to noisy information about the true probability that an event will occur. The security payoff depends on the probability of the event in the manner described below. The information you receive is in the form of biased coin tosses from the true distribution; one thing to be aware of is that the true

distribution can change over time, in the manner described below. Your algorithm has to compete against some other trading algorithms, also described below. We are providing a software framework in Python that will allow you to focus on just writing up your trading strategy and experimenting with it.

2.1 Underlying Model

There are up to R rounds $1 \dots R$, each representing a distinct time period. There is an underlying true probability that the event will occur, which is unknown to everyone, but can change over time. At round i, let p_i be the true probability of the event occurring. During this round i, your trader will be told the outcome of one Bernoulli trial (biased coin toss) with success probability p_i. The trader can then buy and sell the security (there are no inventory or cash restrictions – your trader can take arbitrary positive and negative positions).

2.1.1 Time evolution of p_i

The true probability is a jump process. p_0 is chosen uniformly at random between 0 and 1. At the beginning of a round, p_i is calculated as follows: with probability $1/R$, $p_i \sim N(p_{i-1}, \sigma_{\text{jump}})$, and $p_i = p_{i-1}$ otherwise. That is, at any time, with probability $1/R$ the true probability jumps, and when it does jump, the new true value is drawn at random from a normal distribution centered on the present true value. If $p_i \leq 0$ or $p_i \geq 1$, the security liquidates prematurely at 0 or 100 respectively. Otherwise, the value of the security is $100p_R$ after round R. Note that this is somewhat different from the payoff being 0 or 1 depending on whether the event actually happens or not (think about why it's OK to use this model instead of actually having the event occur with probability p_R and pay off 0 or 1 in a simulation setting like the one here).

2.1.2 Parameters

For this assignment, $R = 100$ and $\sigma_{\text{jump}} = 0.2$.

2.1.3 Competing Algorithms

We have provided code that implements various other types of trading bots. Specifically, there is code for a simple *fundamentals trader*, which receives the same information your bot does and trades in the direction of its belief, which is based on the information it has available to it. There is also code for two different *technical traders*. These are algorithms that do not use information about the fundamental value but instead try to profit from market movements using certain heuristics. In effect, they add noise to the market prices. Details on how to incorporate the different types of traders are in the appropriate readme file – you can control the proportion of fundamentals vs technical traders in the environment, and one of the goals of the project is to analyze performance as you vary this proportion.

The presence of these other traders allows you to potentially glean more information from the market price than would be available to just your own bot. On the other hand, these traders are themselves competing to make money off the market maker, just as you are. Finally, the technical traders add noise to the signal in the market price, but perhaps provide more opportunity for profit!

2.2 Software Framework

We have provided a software framework for you in Python. Please look at it carefully and read the `README` file. The framework handles many tasks, including the generation of data for your bot to use in making trading decisions, handling the market making side of things, and collecting and printing aggregate data from single / multiple simulations.

2.3 Team Composition

You may work in teams of two or three. Please keep in mind that it will be easiest if one of your team members is conversant in Python. Each team needs to hand in only one report (described below).

2.4 What To Do

First, make sure you understand the assignment, and ask questions if you don't, or even if you just want to clarify! Second, start reading and understanding the code, and also start thinking about what kind of trading strategy you want to use

2.5 What To Turn In

We will look specifically for answers to the following questions in your writeup. It is best to address them explicitly. You may add other information that you feel is relevant as well.

1. How does your trader estimate p_i at time i? What algorithm do you use to try and account for the probability of jumps?
2. At which periods does your trader choose to trade? How did you decide this?
3. How does your trader decide what quantities to trade?
4. How do you use the market price in order to improve either your estimate or your trading strategy based on the estimate?
5. Report the mean profit and standard deviation[1] of profits your trader achieves over at least 1000 simulation runs across a variety of different proportions of fundamentals vs. technical traders. Write at least half a page analyzing these results in detail and explaining why you think they come out the way they do.
6. Graphically depict the evolution of prices in one simulation run which you consider typical for your trader.

You will be graded primarily on the quality of your writeup, but also in terms of performance. In particular, strategies that make good average profit without having very high variance of profits will be considered the best.

[1]Controlling the standard deviation is essentially a way of controlling risk. You may be willing to sacrifice some expected profit in order to have less "downside risk" for your strategy. This is particularly important in the real world, where you don't get to run your strategy thousands of times if you run out of money!