Vineet Apte
15-112 F15

Term Project Proposal: PokerBots

For the past few years, MIT has had a course called PokerBots that they offer to their students during their winter term. Teams have one month to program a completely autonomous pokerbot to compete against other teams. As a game of incomplete information and uncertainty, poker is a prime application of the game theory concepts and decision making skills essential to stock trading. Poker players make decisions based on hidden information as well, taking into account factors such as expected value and probability distributions.

My project will address two key issues. First, students can't easily see the poker game that their bot is playing. In other words, there is no 'game footage.' Second, for students who are not enrolled in the MIT PokerBots course, there are no reference players to pit one's bot against. To actually determine if a bot is good, one must test it against a reasonable intelligent player. The PokerBots course only provides one example bot; it plays randomly on every turn.

To address the first issue, I will design and implement a user interface for players to run their bots on the poker engine (currently it is all command-line based) as well as a 'game footage' viewer that plays back the game as an animation. This way, users can see how their bot performs in particular game situations without having to read through hundreds of lines of raw text output.

To address the second issue, I will write at least one "mediocre" bot that can consistently beat the random bot, and one "smart" bot that can consistently beat the mediocre bot. By consistently, I mean that the superior bot should in at least half of all played games so its long-run expected value against the inferior bot is positive.

The MIT Pokerbots engine uses as an packet communication protocol to communicate with the bot. The engine sends packets about the game state via a socket, and the bot can respond with its own packets that contain instructions for player actions. Since the engine and bot are being hosted on the same computer, we use port 3000 for all communications. When the engine is started, it waits for the bot to connect to the local host.

Both bots will use a hand evaluation engine that estimates the pot odds for a particular hand with Monte Carlo simulation. The "mediocre" bot will follow a series of simple, tunable rules that do not adapt from turn to turn, whereas the "smart" bot will have a significantly more complicated strategy. Once I am able, to pit the "mediocre" bot against the "smart" one, I will write a machine learning routine that will collect data on the "mediocre" bot and train itself to predict the "mediocre" bot's actions (techniques to use: classification (aggressive vs. conservative), regression).

List of necessary modules/technologies:

- MIT Pokerbots Engine (.jar file)
- bash files (to execute, python scripts and .jar)
- Sockets (local host connection)
- itertools (to compute pot odds)
- tkinter
- os (to write to a configuration file)
- Monte Carlo simulation (will use random module)
- scikit-learn
    - It is unlikely that I will use scikit-learn for my project. Thus, I will not include it in my feasibility demonstration.