Assignment: Pitch Class

Introduction to Software Engineering CSCI 2360

Objectives

- Learn how to run the Pitch program, understand the output, and analyze play.
- Learn how to use scenarios to define play strategies.
- Learn how to create/modify scenarios to change play behavior.
- Begin devising your bid and play strategies for the end of semester Pitch tournament.

Overview

The purpose of this assignment is to gain experience analyzing and specifying play strategy using scenarios. This understanding will enable you to further refine your bid and play strategies for the Pitch Tournament at the end of the semester. It will likely be useful to use the `Pitch Play Strategy` presentation as a reference for this assignment.

Background

Executing `pitch.py` in a shell

In the examples below, `>` represents the shell command prompt. Depending upon your operating system and configuration, your command prompt may look different from this.

After opening a shell, navigate to the directory holding your code. This will likely require using the `cd` (change directory) command to navigate the file system. For example, if your code is in the `pitch_student` subdirectory within your `source` directory, you might use the following command to change to that directory.

Here is an example on a Mac.

```
> cd source/pitch_student
```

Windows uses a backslash (`\`) rather than a forward slash, so the following example is for Windows.

```
> cd source\pitch_student
```

To run `pitch.py` using the following command. By default, this plays a single game with default strategies for 4 players.

```
> python pitch.py
```

The `pitch.py` file has several command line options. Use the following command to see the help information for the options.

```
> python pitch.py --help
```

Executing `pitch.py` with a configuration file

The file `wizards_equal_config.yaml` is a sample configuration file for `pitch.py`. This file uses YAML (Yet Another Markup Language) as the syntax to specify attributes about players. Use the VS Code editor to see the contents of the configuration.

There are 4 players configured. Each player has a `name` and `strategies` attributes.

In this configuration, all players are set to the same bid and play strategies (hence the "equal" in the configuration file name). Each player uses the `BaseScenarios.xlsx` file for bid strength values and play scenarios. Each player uses the `PlayerDecisionTree` class so their "bot" can use scenario based play strategies.

Here is an example of how to run `pitch.py` specifying the `wizards_equal_config.yaml` configuration file.

```
> python pitch.py -c wizards_equal_config.yaml -i -v -d
```

For a little less detailed output, try the following command.

```
> python pitch.py -c wizards_equal_config.yaml -i -v
```

Task Description

Analyze the default configurations

- Run several instances of `pitch.py` using the sample configuration and analyze the bids and play strategies produced for different dealt hands.

Make your own strategy file

- Copy the `BaseScenarios.xlsx` file to a new file. Name your scenario file with your chosen team name. For these instructions, we will call it `Hogwarts.xlsx`.

- Make a new sheet in this file with a distinct name and copy your bid strength configuration from the previous assignment into this sheet. For example, you might name the sheet `FirstYears` and copy your bid strength configuration into this sheet.

- You are going to also modify the scenario definitions in your new strategy file, but that will be a later step.

Make your own configuration file

- Copy the `wizards_equal_config.yaml` file to a new file. Pick a name for your configuration file. For these instructions, we will call the configuration file `hogwarts.yaml`.
- Edit your configuration file (e.g., `hogwarts.yaml`) so that your name and a teammate's name replace one of the existing teams. For example, you can replace Harry and Ginny with your team.
- Change the `bid_strength_file` and the `scenerio_file` for both players on your team to your new scenario configuration file (e.g., `Hogwarts.xlsx`). Also change the `bid_strength_sheet` for the players on your team to your new sheet (e.g., `FirstYears`). Note that the players on your team need not have the same strategies, but for starters you may choose to make them the same.
- Change the `aggressiveness` and `restraint` values for the players on your team to values you think appropriate. The players on your team need not have the same values for these strategies.
- Leave the configurations for the other team (e.g., Ron and Hermione) as is. This team will be your opponent and play the strategies already defined for them.

Confirm your configuration and strategy files

- Run the Pitch program several times with your newly created configuration and strategy files. Analyze the play of your team versus the existing team.

  For play analysis, you may wish to run a single game at a time in "interactive" mode. The command to do this will look something like this with the appropriate configuration file.

  ```
  > python pitch.py -c hogwarts.yaml -i -v
  ```

- To see how your current team configuration would fair against the existing opponents, run a match with several games. The example below runs 101 games and reports the results. Note that in this example, the `-i` and `-v` command line switches are not set. Run several matches to assess how your team is competing.

  ```
  > python pitch.py -c hogwarts.yaml -g 101
  ```

Assess your bid strategy

- At this point, the only difference in play between the two teams is likely just their bid strategies. Try different configurations for your bid strengths, aggressiveness, and restraint. Run multiple matches with each trial configuration and adapt your bid strategies as you see fit.

Assess your new play strategy

- Look at the `Position_1` sheet in your scenario definition file (e.g., `Hogwarts.xlsx`). Using this scenario defintion, a player playing in the first trick position and who also has the "boss" (i.e., highest remaining trump card), will execute the action `play_highest`.

You can confirm that this is actually works by executing the test hands `ace_all_trumps` and `ace_not_all_trumps` as shown below (note the double dash "--" on the `--test_hand` switch).

```
> python pitch.py -c hogwarts.yaml -v -d --test_hand ace_all_trumps
```

And...

```
> python pitch.py -c hogwarts.yaml -v -d --test_hand
ace_not_all_trumps
```

- Modify the `Position_1` scenarios so that a player using this set of scenarios executes `place_highest` if they have the "boss" and are all trumps, but executes `play_off` if they have the "boss", but are not all trumps.
- Verify that your modified scenarios execute the desired strategies by running the test hands above with your new configuration. Capture your output into the files `test_ace_all_trumps.txt` and `test_ace_not_all_trumps.txt` using something like the example commands below.

  If you are on a Windows machine, first execute these commands to make sure you can successfully redirect your output to a file.

```
> chcp 65001
> set PYTHONIOENCODING=utf-8
```

  Now...

```
> python pitch.py -c hogwarts.yaml -v -d --test_hand ace_all_trumps >
test_ace_all_trumps.txt
```

  And...

```
> python pitch.py -c hogwarts.yaml -v -d --test_hand
ace_not_all_trumps > test_ace_not_all_trumps.txt
```

- Submit your work by committing all your work to your GitHub project.

Submission Requirements and Grading

- Your new configuration file properly created, updated, and committed with all changes to names, strategies, etc. as described above. (5 points)

- Your new strategy spreadsheet properly created, updated, and committed. The weights submitted must be modified from the sample values and the "Position_1" scenarios modified per the instructions. (10 points)
- `test_ace_all_trumps` and `test_ace_not_all_trumps` properly created and committed. The output must demonstrate successful execution with your new new scenarios. (10 points)
- Submit a link to your GitHub repository on Blackboard.

Due Dates

The due date is specified on Blackboard.

Remember the Pitch Tournament!

Remember, there is an awesome Pitch Tournament at the end of the semester!

Start working on your game strategies now! Analyze your play strategy across multiple iterations. Adapt your scenarios as you see fit. Improve your game!

There are fabulous prizes! Whose team will win?

Good luck! Remember to check the requirements and expectations carefully and ask questions if you need clarification.