Bryant Bettencourt
Spring 2020

Calday Tournament Landing Page, Spring 2020

Team: Front end
Sub-team: Calday Tourney Sp20

Summary:

I created the landing page for the CalDay tournament this semester. The project can be described in three parts: creating the actual landing page to host different GCUni games, hacking the GCUni pages to work with the tournament page, and collecting and analyzing the gathered data.

Tournament Landing Page:
Link: https://nyc.cs.berkeley.edu/~bryant/GamesCraftersTourney/parent.html

This page was specially created for the Calday Tournament. Currently it is hosted on my personal directory on the GCUni website. The page is fully functional, but needs some extra touches if we want to host it in a more permanent way. It is unclear how this page might be used for the future, however, it can be used again for another calday or tournament event.

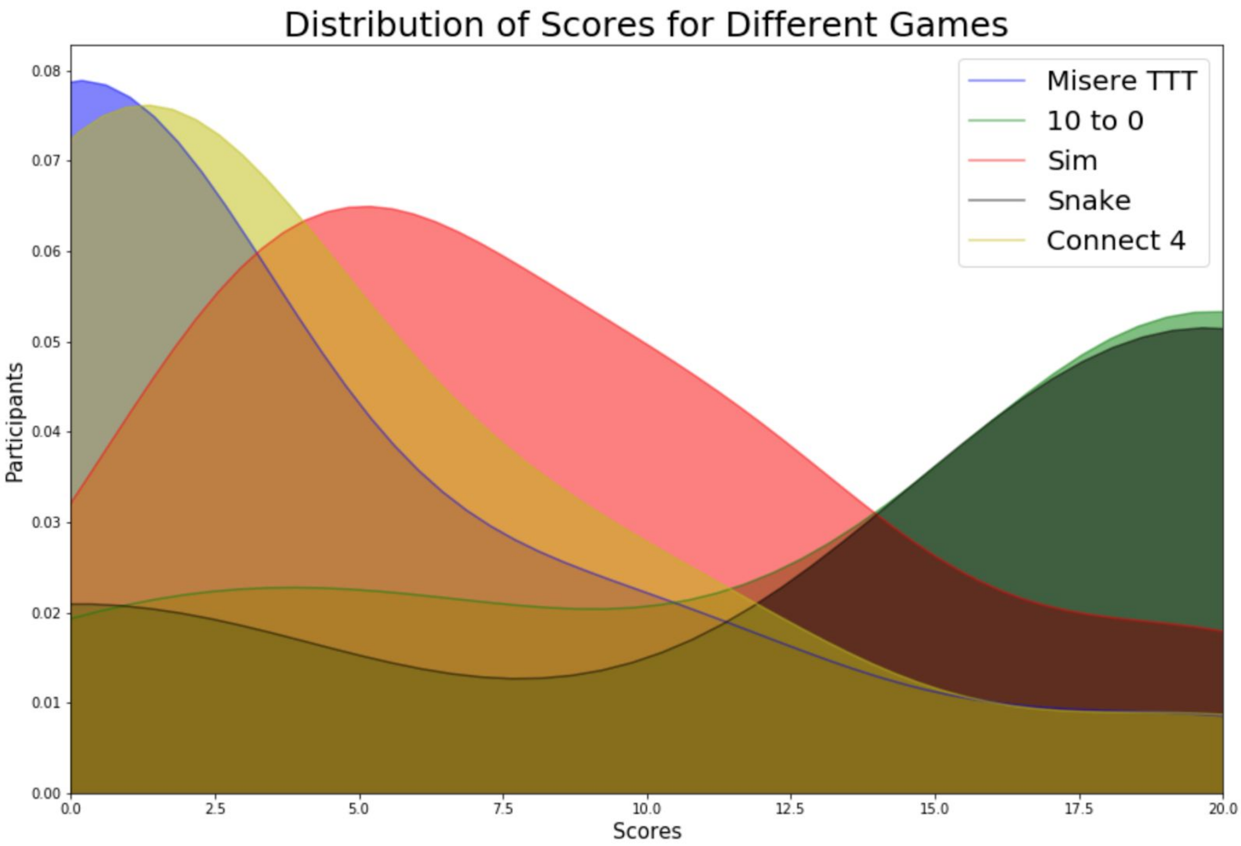
Making the tournament landing page involved classic web page-building tools, including html, javascript, and css. The main functionality of the website was to be able to play games hosted by gamesman uni, record the data from those games, and send them to a backend server for analysis.

GCUni Hacking
Link: https://github.com/GamesCrafters/GamesmanUni/tree/bryant

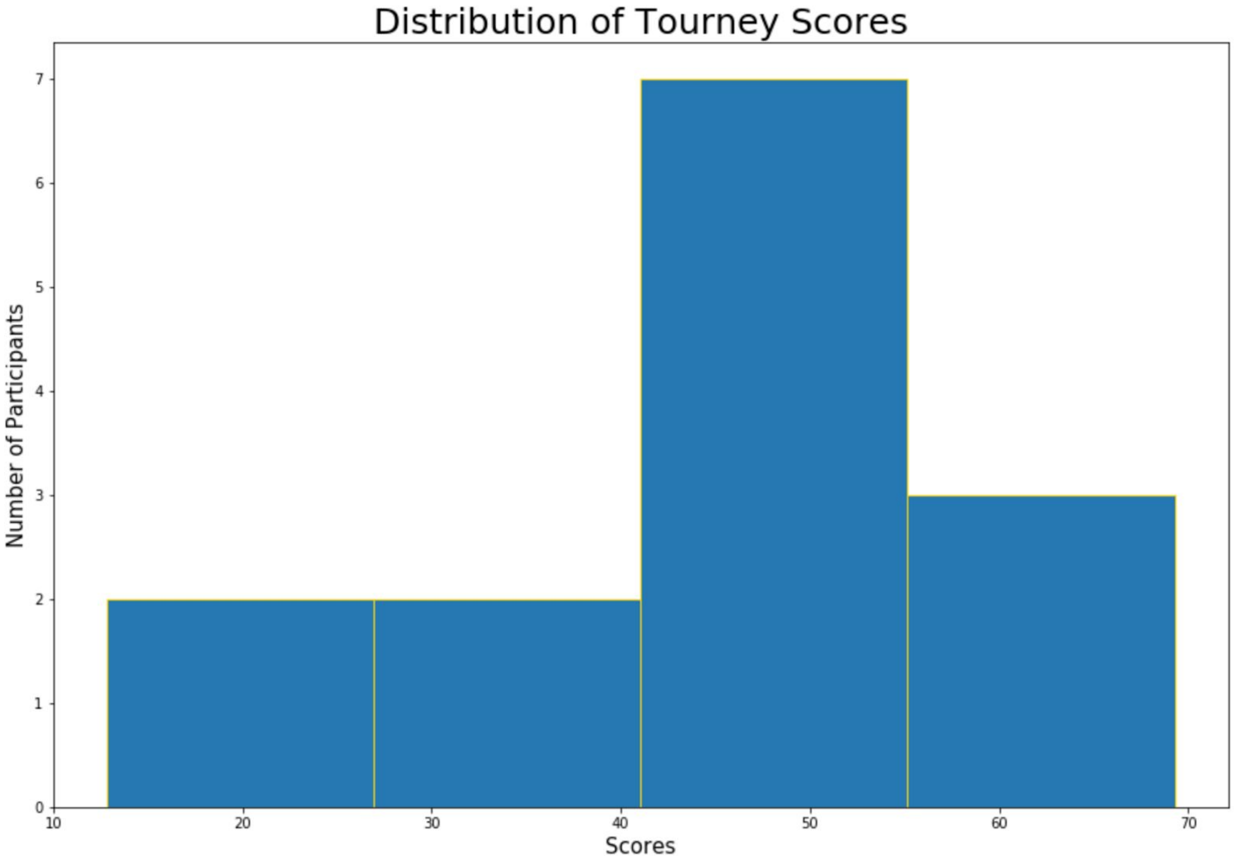
Most of the hacking of Gamesman Uni was done by Seth Lu, I contributed by helping to figure out how to give the Gamesman Uni pages the ability to communicate with a parent iframe. The start of this part of the project was a branch of Gamesman Uni. Currently my branch of Gamesman Uni contains the modified code that was made for this project. Currently the communication only works properly if it is one of the 5 games we chose: misere tic tac toe, snake, sim, connect4, or 10 to 0.

Move Analysis:

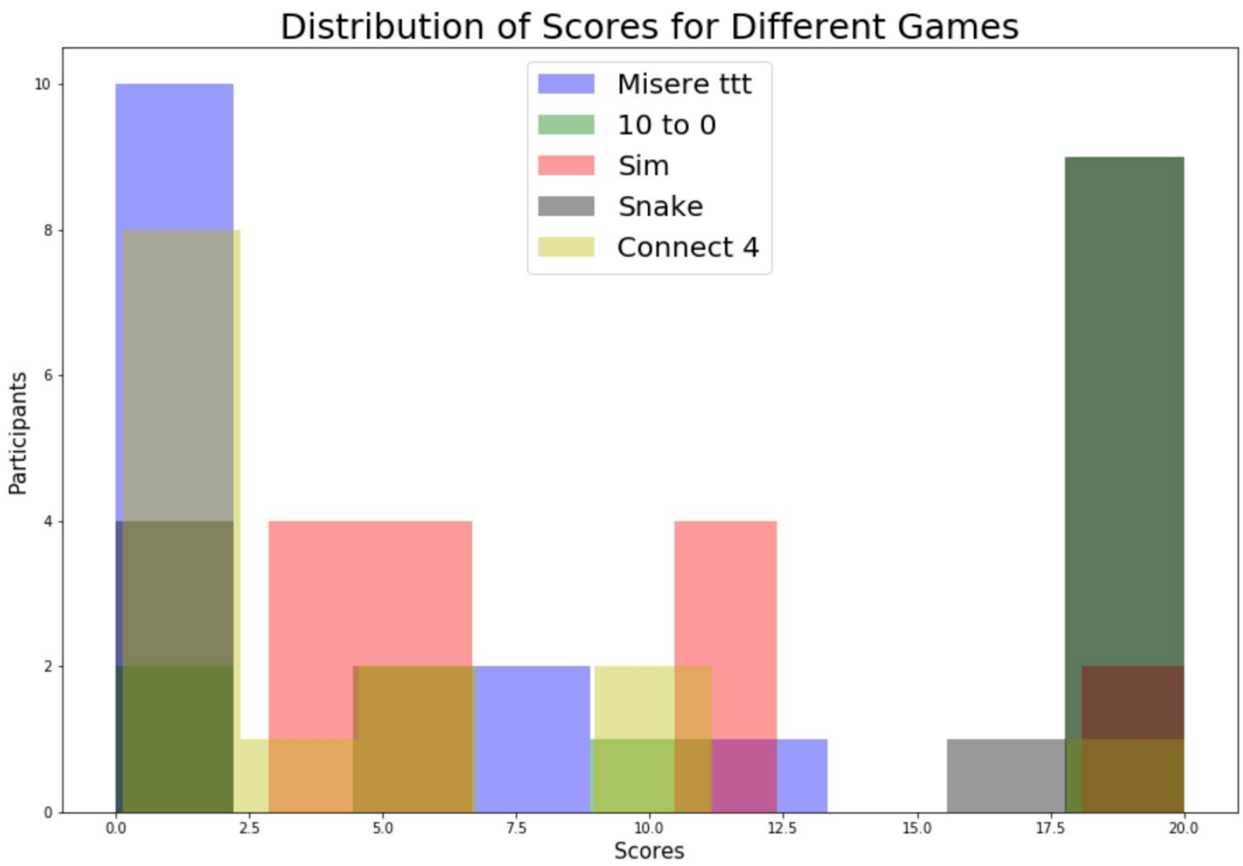
Move analysis was done by me using Jupyter Notebook. The data for all of the data gathered by the tourney website was sent in json form to a backend server hosted on nyc.cs.berkeley.edu under the filepath: `/var/www/html/calday2020/tournament/submissions.` For each game, along with the moves made for each game, the values of each move they made were also sent to the server. With this info, I was able to create a few graphs using Matplotlib, Numpy, and Seaborn plots. In the future, I hope to be able to gather a larger amount of data and refine the scoring system. Right now there are a few problems, like not rewarding players for beating the game in less than the expected number of moves.


Distribution of Scores for Different Games
Misere TTT
10 to 0
Sim
Snake
Connect 4
Participants
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Scores

Bryant Bettencourt
Spring 2020



Bryant Bettencourt
Spring 2020


Distribution of Scores for Different Games
Misere ttt
10 to 0
Sim
Snake
Connect 4
Participants
Scores