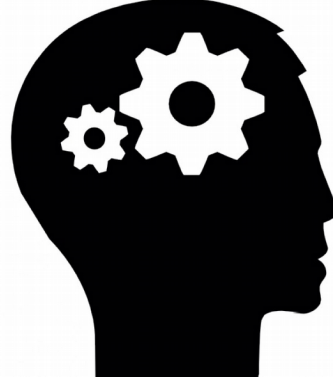


# UFO GAME

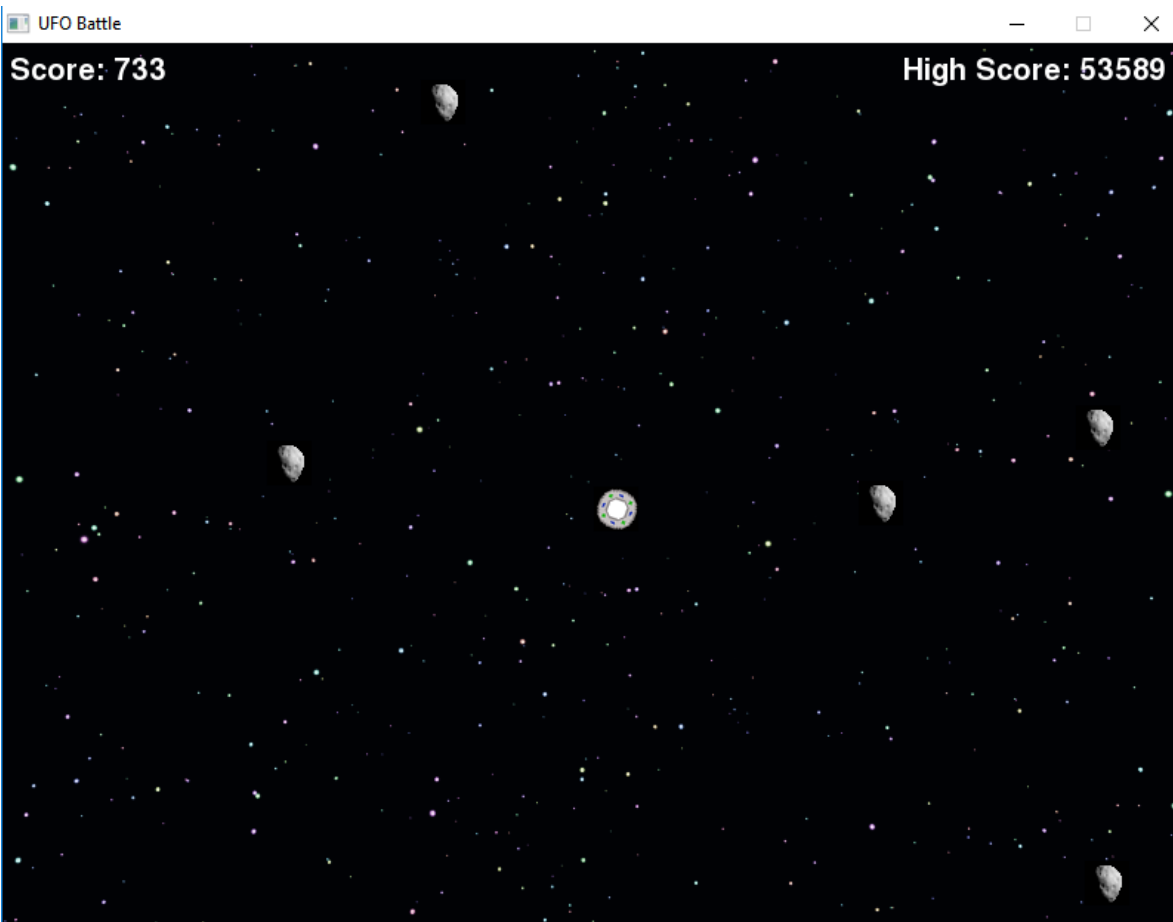
## Dodging Asteroids with Machine Learning



Zachary Neubert  
CS 5600 Fall 2016

**Abstract:** The purpose of this project is to test different methods of machine learning and different features to play a real-time obstacle avoidance game. Many methods have been tried, including neural networks and several different types of classifiers. Different methods of feature extraction are also shown.

**Problem Description:** UFO Game is a game written in python where the player uses the arrow keys to control a UFO in a window. The objective is to avoid collisions with asteroids for as long as possible. Both the UFO and the player bounce off the edges of the window, conserving all of their momentum. Asteroids also bounce off of each other. The game awards a point for every frame that the player survives.





Miscellaneous Stats:  
Laptop Memory Failures: 2  
Total Truth Files Generated: 62  
Total Truth Data Generated: 13.4 GB

**Method:** Because I had direct access to the game's source code, I decided to build the learning directly into the game. I refactored the keystroke reading to allow keystrokes from other sources. Then, I created a "random" player for the game, that inputs up, down, left, right, or nothing randomly every frame.

I then implemented controls for many different classifiers. I used several classifiers from Sci-Kit Learn, as well as the Keras wrapper for tensorflow neural networks to test this.

**Feature Extraction:** Because I had access to the source code for this game, I was able to extract the features directly from the objects. I had a few different sets of features to try. The first one was the x and y locations and velocities for both the UFO and the asteroid.

The second set of features I tried was an attempt to make simpler, easier to understand features. So I generalized the features into a grid:

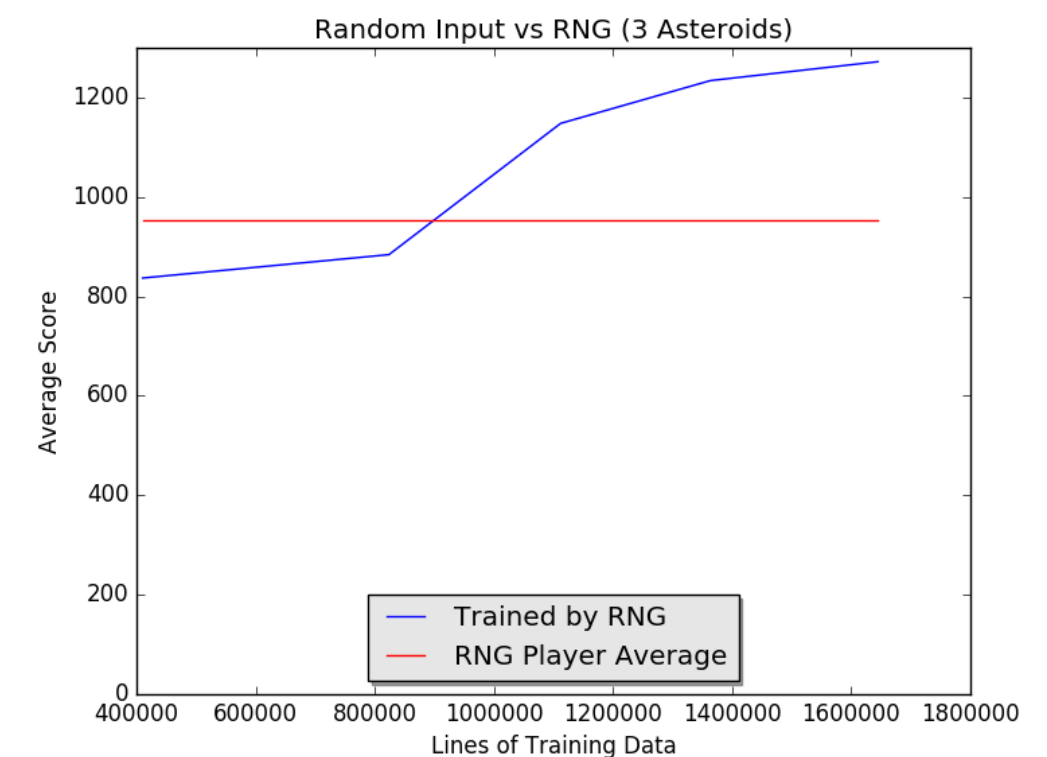
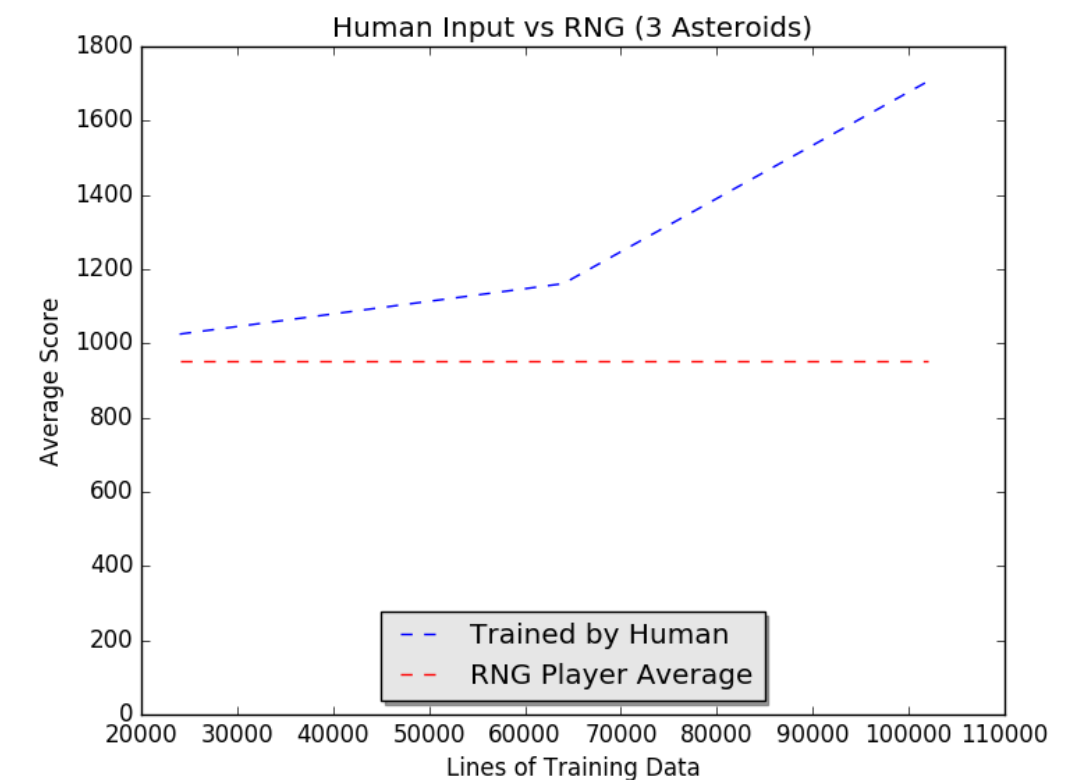
0	0	0	0	0	0	0	0
0	0		0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0		0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

To teach the classifiers to win, rather than to imitate a random number generator, all frames within 150 frames of dying were forgotten. Because there are far more good moves than bad in a 1-asteroid scenario, it should be most important to learn the moves that do not kill the player.

**Results:** The Random Forest classifier and the Kernelizer from sci-kit learn did fairly well, but the Random Forest did better, so I focused on training that.

The second set of features also did far better than the first.

Human input was orders of magnitude better than Random input for training. However, it also took far longer to create. Here are the results for the 3 asteroid scenario, organized by the amount of truth data used.



For more results, visit:

<http://www.github.com/ZacNeubert/UFOGame>