Randomized Optimization

Introduction

For the first portion of this assignment, we are to apply four different search techniques to three different optimization problems. Those four search techniques are randomized hill climbing (RHC), simulated annealing (SA), a genetic algorithm (GA), and MIMIC. The first problem will highlight advantages of GA, the second will highlight advantages of SA, and the third will highlight the advantages of MIMIC. The parameters tuned for each search technique can be seen in the table below.

Randomized Hill Climbing	Restarts = [10, 25, 50, 100]
Simulated Annealing	Temperatures = [2, 4, 6, 8, 16, 32, 64, 128, 256, 512, 1024, 2048
Genetic Algorithm	Population Size = [50, 100, 150, 200] Mutation Rate = [0.2, 0.3, 0.4, 0.5]
MIMIC	Population Size = [50, 100, 150, 200] Keep Percentage = [0.01, 0.02, 0.04, 0.08, 0.16, 0.32]

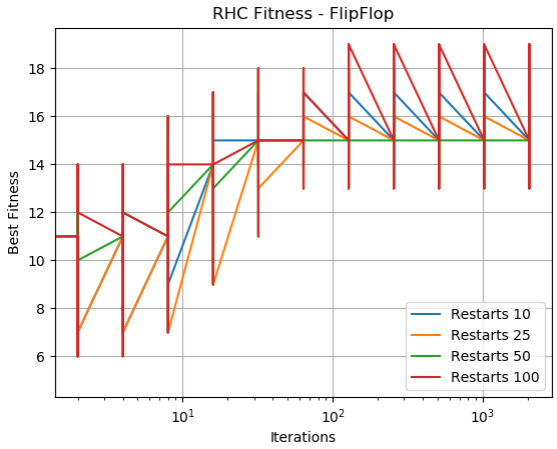
For the second portion of the assignment, we will use RHC, SA, and GA to find good weights for a neural network and compare our results to those seen in Assignment 1.

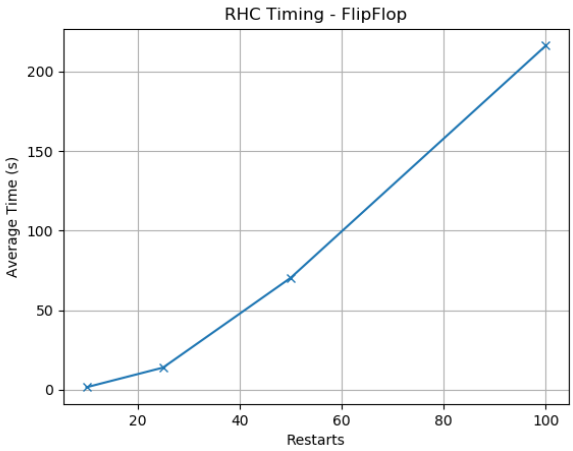
Problem 1

The first problem chosen is the Flip Flop problem. The problem is about maximizing the number of bit alterations in a bit string between 0 and 1. Every algorithm used for this problem did reach the max score possible, 19 although some performed much faster and with less iterations.

Randomized Hill Climbing

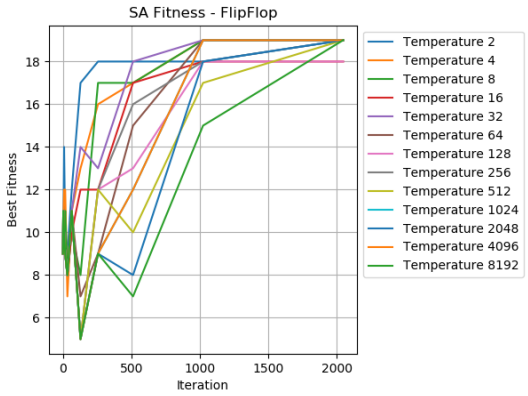
Randomized Hill Climbing reaches the optimal score, but only with a lot of restarts over more than 1000 iterations. The time needed to reach the best score ends up taking orders of magnitude longer than the other search techniques. I ended up adding more restarts when it looked like it would increase the fitness, but the time needed to run the tests soon became too much, so I capped it off at 100.

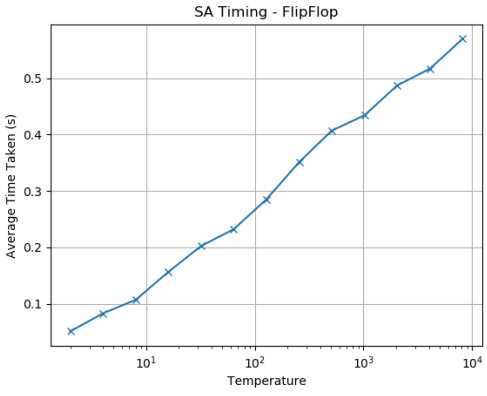




Simulated Annealing

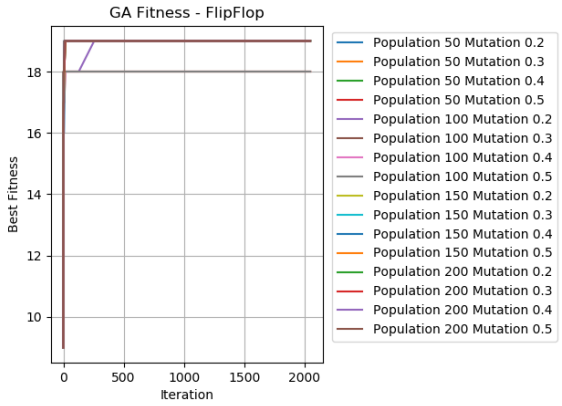
SA reaches the optimal score, but it takes hundreds of iterations to do so. It does find the optimal score in a much quicker time than RHC does, though.

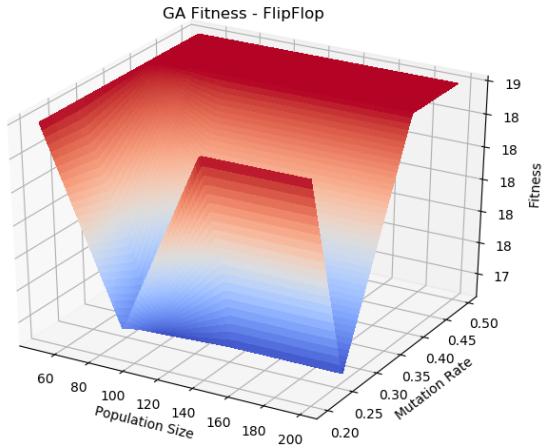


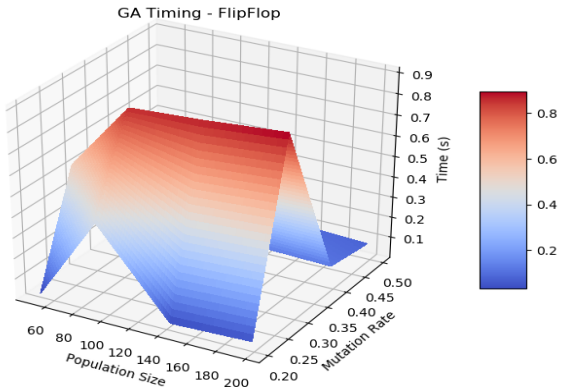


Genetic Algorithm

The GA search technique performed the best in this case, reaching the optimal score with the least iterations needed and in a quick amount of time.

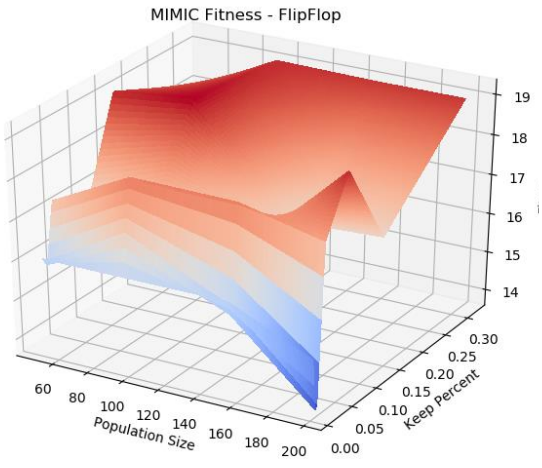


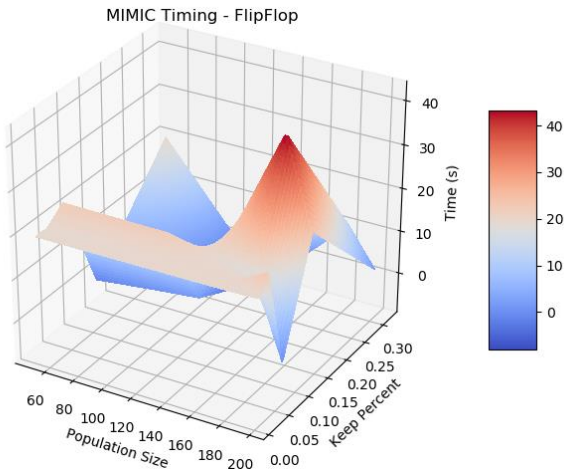




MIMIC

The MIMIC algorithm performs well, but in not as fast of a time as the GA.



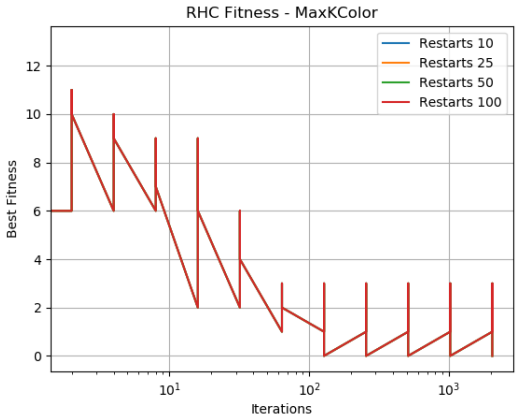


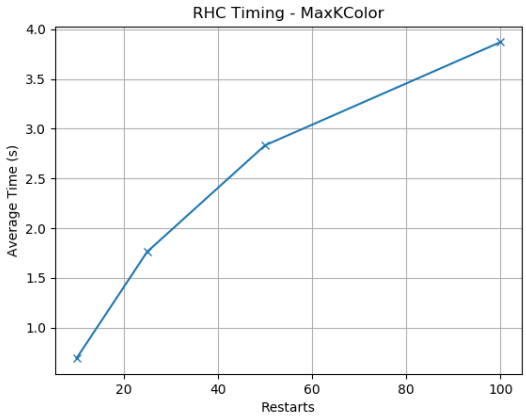
Problem 2

The second problem chosen is Maximum K-Coloring problem. The problem is to color a maximum number of vertices using k colors with the goal being that no two adjacent vertices have the same color.

Randomized Hill Climbing

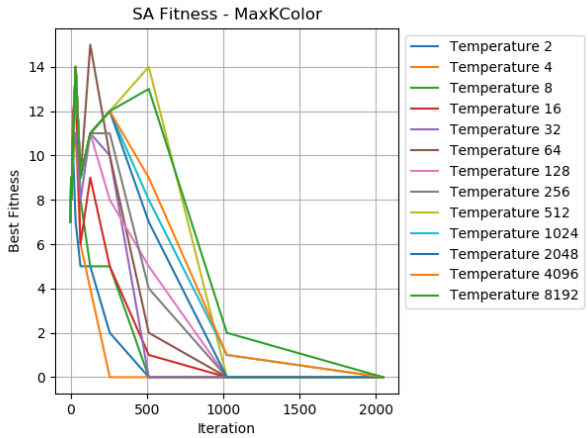
RHC does not perform well on this problem, and for some reason seems to do worse with increasing iterations. If given more time to work on the project, I would investigate why the algorithm seemingly does worse with more opportunity to learn.

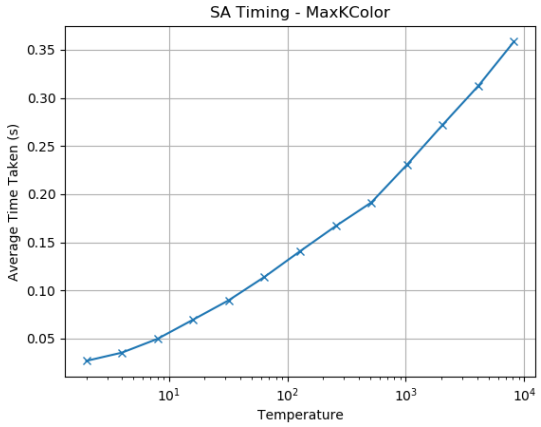




Simulated Annealing

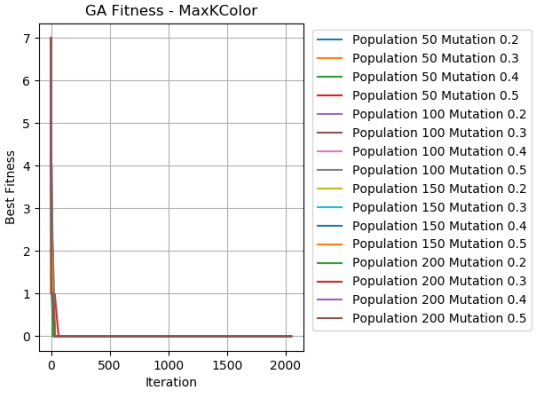
SA does the best on this problem, reaching a high fitness score with minimal iterations needed. Although the time is increasing exponentially with a higher temperature value, it still takes a minimal amount of time to reach a solution.

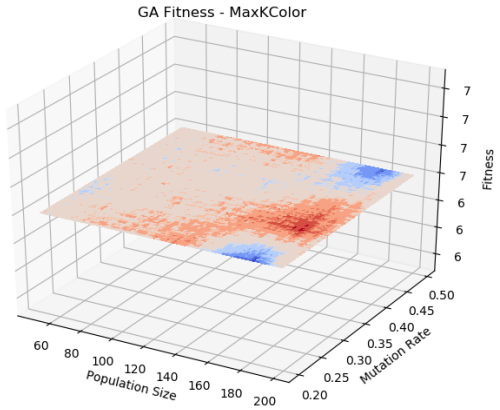


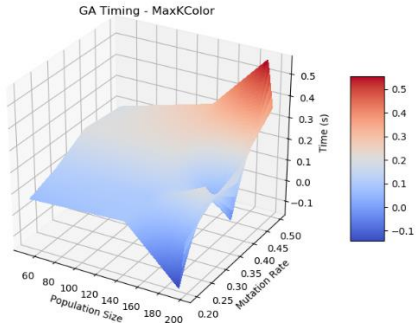


Genetic Algorithm

The GA technique does not perform well on this problem. It, along with MIMIC, greatly underperformed RHC and SA, although it was faster in reaching a solution than RHC. As can be seen by the fitness surface plot, the fitness score doesn't seem to even change with regards to population size of mutation rate, which is interesting.

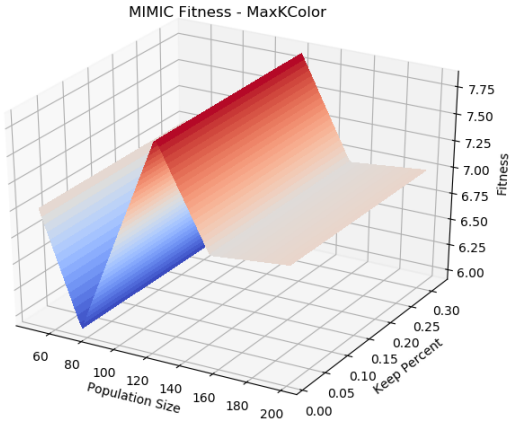


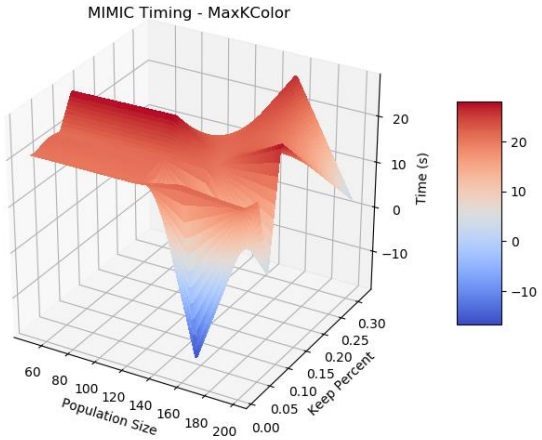




MIMIC

MIMIC performed a little better than GA for this problem, but still not good compared to SA, with added time needed.



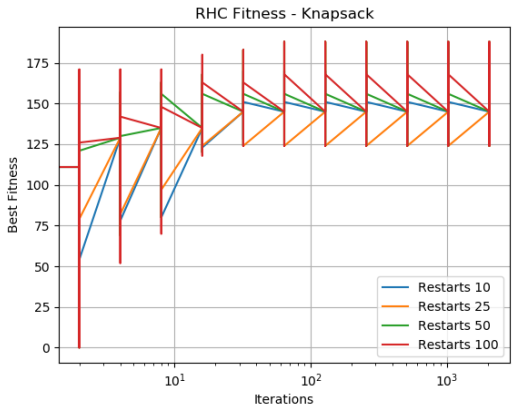


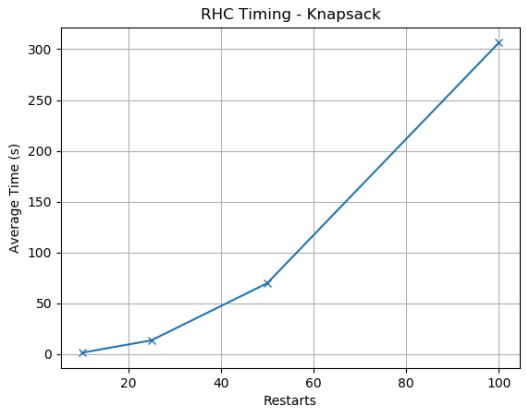
Problem 3

The third problem chosen is Knapsack. Knapsack is a NP-hard constrained combinatorial optimization problem where when given a set of objects, their weight and corresponding value, the goal is to determine the number of each object to include in the "bag" such that the total weight is constrained within a given number and the total value is as large as possible.

Randomized Hill Climbing

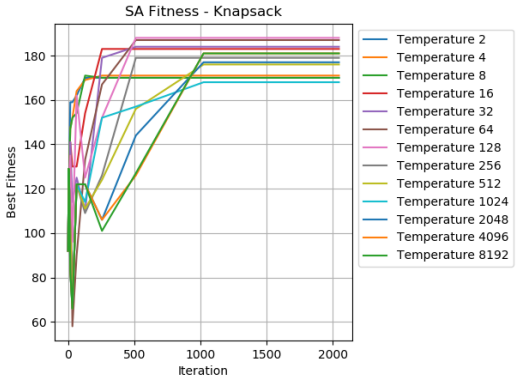
The number of restarts doesn't seem to help too much with this problem. This is likely because there are too many local optima present with the number of variables that are involved. The time taken to run this algorithm increases exponentially with the number of restarts involved.

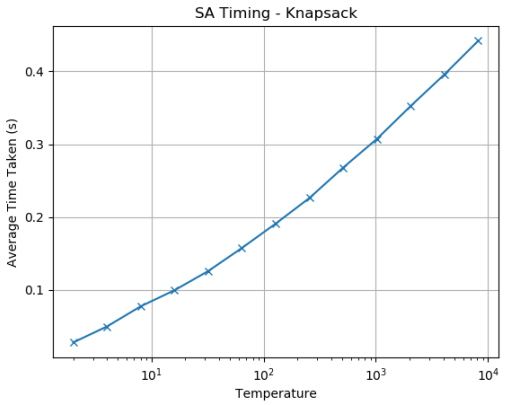




Simulated Annealing

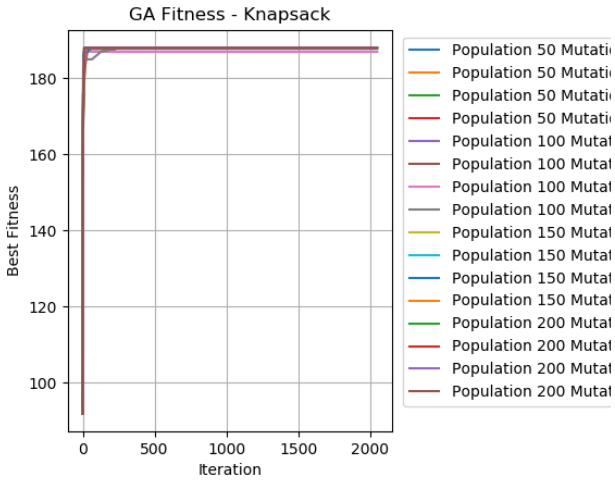
The SA search technique again runs fairly quickly, but the fitness is not as good as other search techniques such as MIMIC and GA.

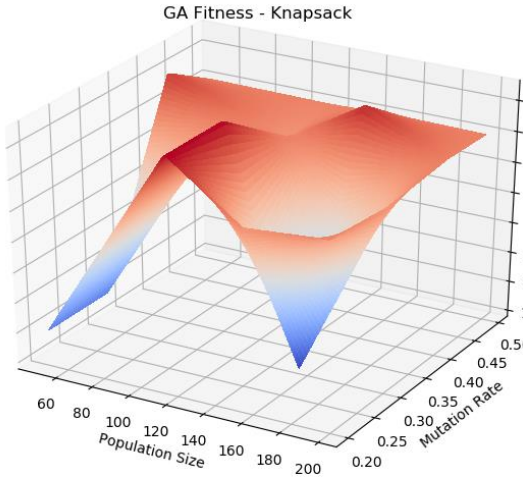


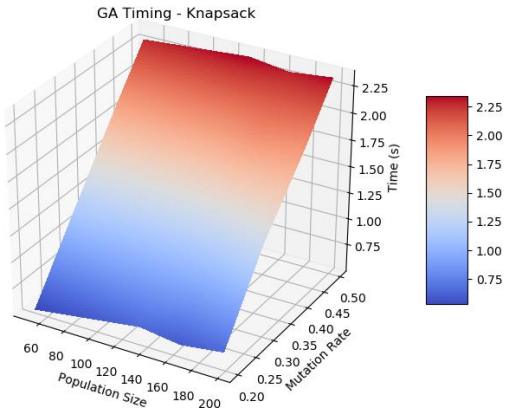


Genetic Algorithm

The GA algorithm performs well in the Knapsack problem, with the time of evaluation increasing linearly with the mutation rate.

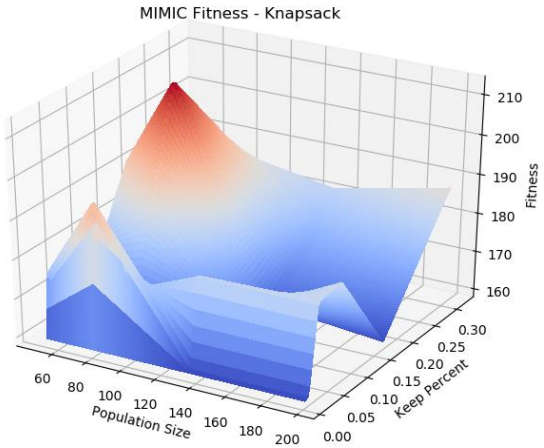


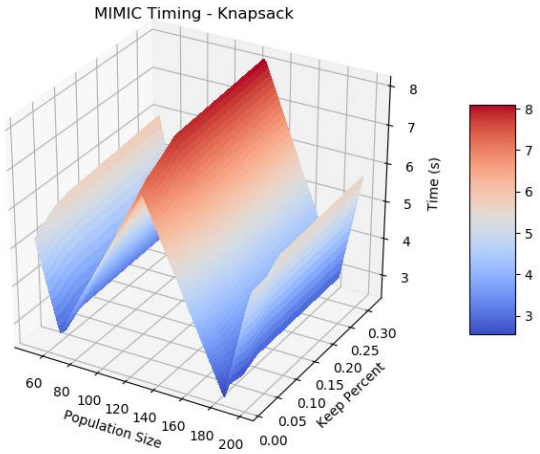




MIMIC

MIMIC performs best on the Knapsack problem due to the nature of the problem. It benefits from the structure involved with the problem and can retain history, unlike the RHC and SA algorithms.

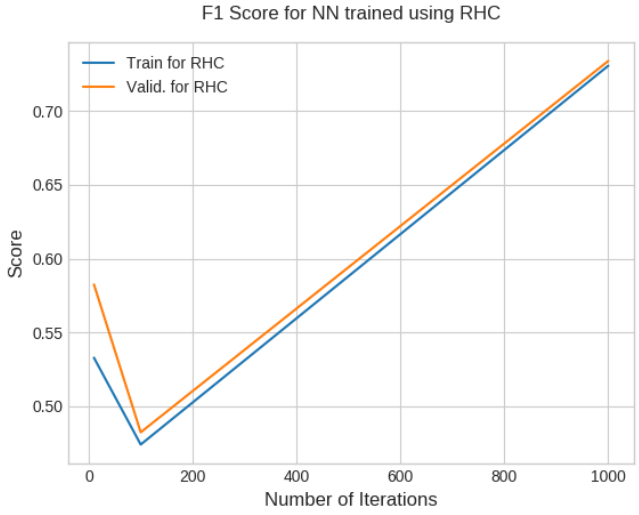




Neural Network

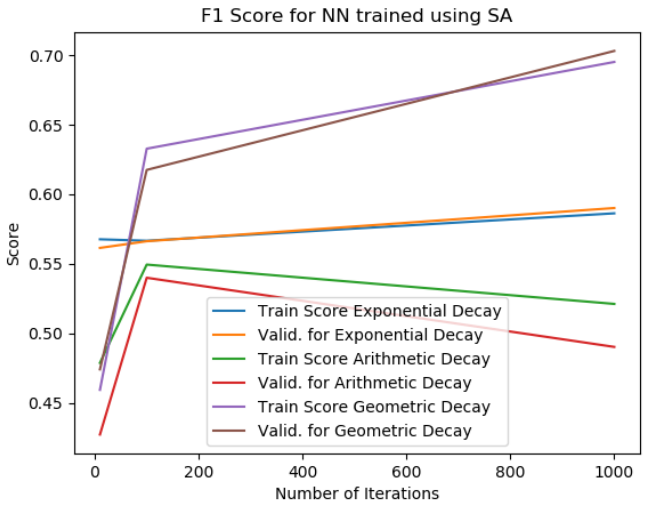
The goal for this part of the project was to use three of the search techniques (RHC, SA, and GA) to attempt to optimize the weights of a neural network as opposed to using back propagation like in Assignment 1.

Randomized Hill Climbing



Simulated Annealing

Unfortunately, the plot got lost in transferring data to a different computer, but the genetic algorithm approach actually had to best results in the end, with the least amount of error for both the training set and test set.



Genetic Algorithm

Unfortunately, the plot got lost in transferring data to a different computer, but the genetic algorithm approach actually had to best results in the end, with the least amount of error for both the training set and test set.

Analysis

Previous Results

For Assignment 1 with supervised learning, the neural net approach did well using grid search, as can be seen below.

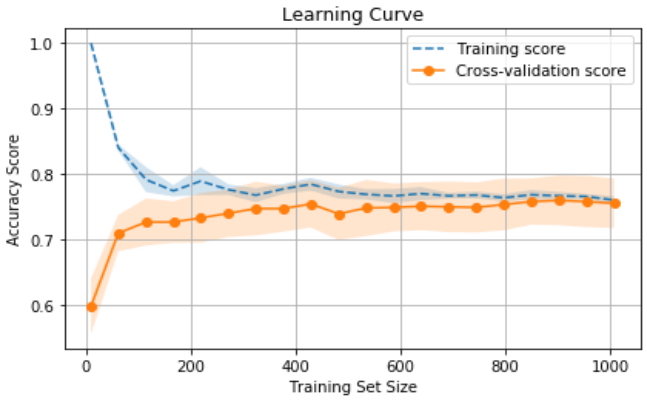


Figure 1: Learning curve for Neural Network on wine quality dataset

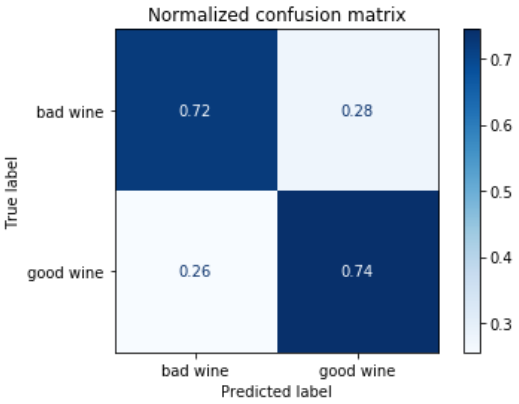


Figure 2: Confusion Matrix for Neural Network

New Results

For using randomized optimization to try and calculate the weights that went into the Neural Network, two of the three search techniques turned out with worse score than were previously achieved. If given more time, I would broaden the search space of parameters that were tested.

Search Technique	*F1 Score*
Randomized Hill Climbing	0.4166
Simulated Annealing	0.438
Genetic Algorithm	0.7566