Assignment 2

(svyas44)

I. Optimization Problems and Results

In the first part of the experiment, it was asked to identify 3 non-trivial optimization problems to highlight advantages of Simulated Annealing (SA), Genetic Algorithm (GA) and MIMIC w.r.t each optimization problems.

Experimental Design: In this experiment, we identify and build 3 optimization problems as listed in the next section. We run all 4 algorithms on each of the problems and compare different algorithms based on multiple parameters such as Clock Time, Best Fitness Value, Number of Function Evaluations per sec, Convergence, etc. We present the findings in Graphs and Tables.

A. OneMax: SA / RHC

The OneMax optimization problem is a simple binary optimization problem where the goal is to maximize the number of ones in a binary string of fixed length. Each solution for the problem is listed as a binary string of fixed length and the fitness is calculated by counting number of number of ones in Binary string. The OneMax fitness function has an interesting pattern of many Local Optimas – each corresponding to a Binary string with all ones. This provides optimization algorithms to get trapped in suboptimal solutions. The search space also grows exponentially as the length of the string increases. This makes a suitable problem to measure scalability and performance of different optimization algorithms.

My initial hypothesis was that this problem seems like a perfect candidate for evolutionary algorithms like GA and MIMIC. As the candidates keep on growing, the algorithm should be able to able to search over multiple suboptimal candidates. However, since the fitness function has a single peak, Randomized Hill Climbing might also serve as a very good candidate considering a single Global optimum.

Information Classification: GENERAL

Results in Plots:

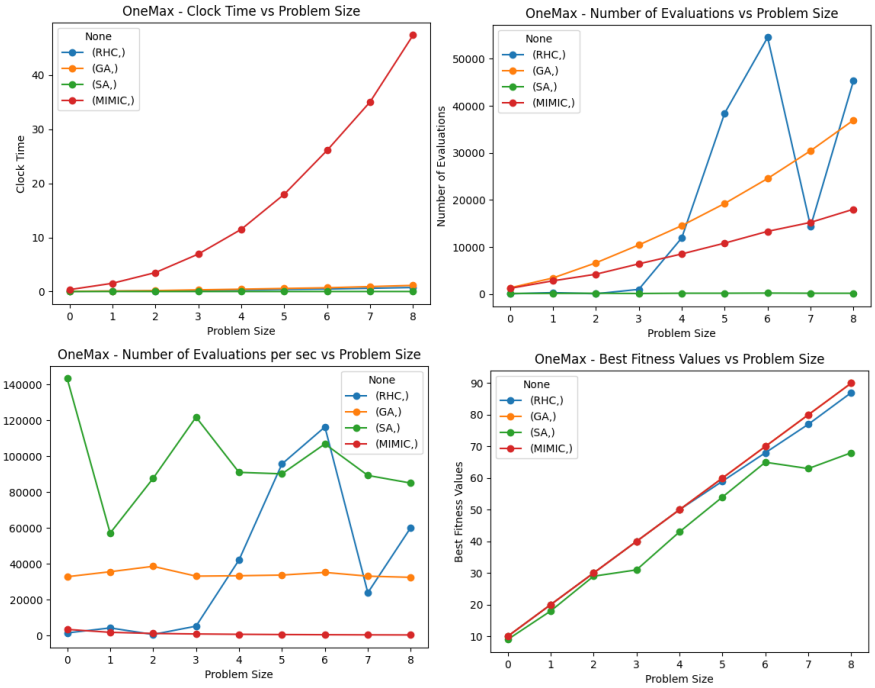

Figure 1: Comparison of Multiple Opt-Algorithms on OneMax problem.

Based on initial Hypothesis, SA and RHC - both seem to perform good if we compare with others. Both achieve quite high fitness (comparable to Advanced algorithms like Genetic and MIMIC with very high rate of Function Evaluations per sec) – and reach convergence faster.

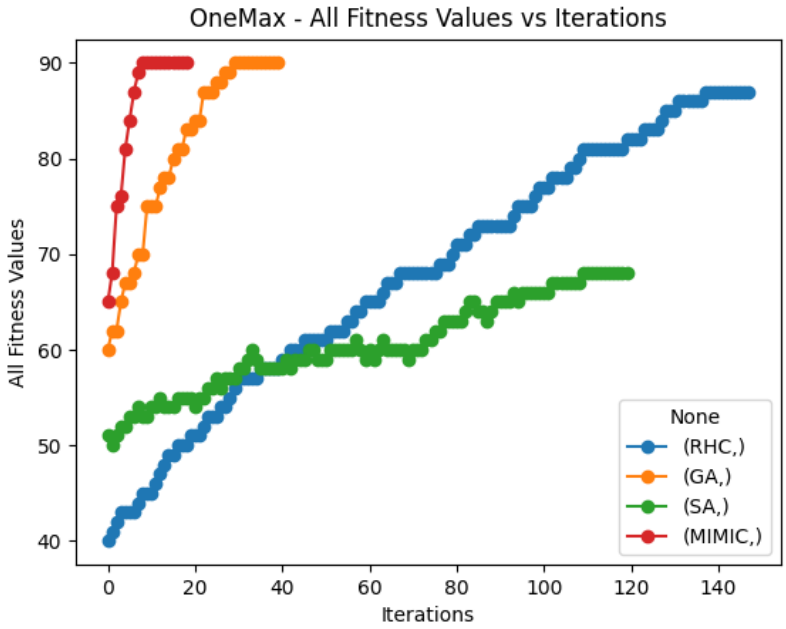

Figure 2: Convergence Plot for multiple algorithms

Information Classification: GENERAL

B. Knapsack: GA

The Knapsack Optimization problem involves maximizing the value of items that can be places into knapsack with a limited weight capacity. Each item has a weight and a value, and the goal is to maximize the total value by selecting a subset of items. But the total weight shouldn't exceed the Knapsack capacity. This problem involves optimizing the combinations required to meet the knapsack capacity. This problem is not like the continuous optimization problem with smooth, continuous fitness landscapes (like the previous one).

Genetic Algorithms might perform better since one can argue that, based on few candidate solutions, the algorithm can generate new solutions using crossover and mutations to come to an optimal solution. The ability to mutate and propose amongst constraints can be a good feature for Genetic Algorithms.

Results in Plots:

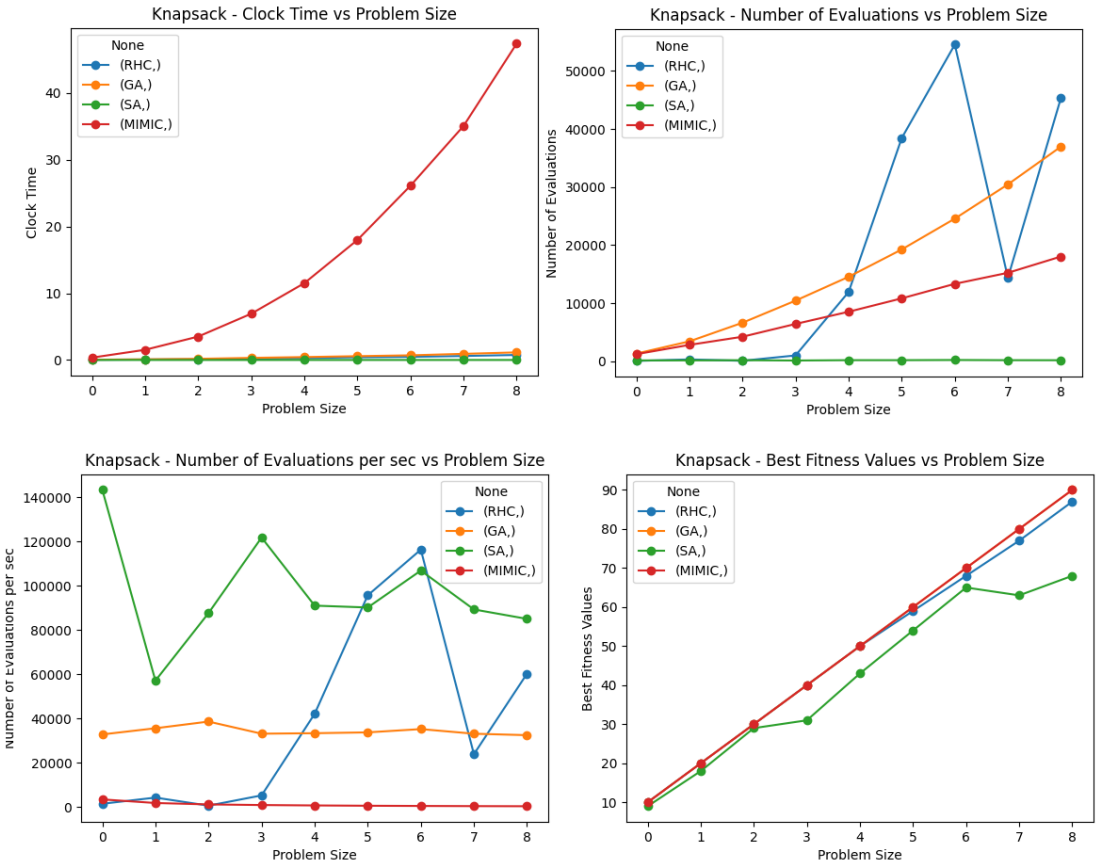

Figure 3: Comparison of Multiple Opt-Algorithms on Knapsack problem.

For Knapsack problem, GA seems to perform consistently well for getting high fitness values for given problem size. It also reached convergence faster as compared to other algorithms as well.

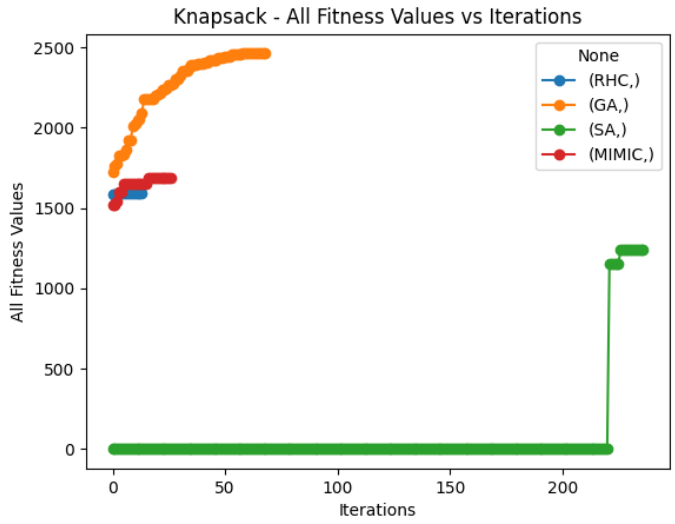

Figure 4: Convergence Plot for Knapsack Problem

C. Travelling Salesman (TSP): MIMIC

The goal with TSP problem is to find the shortest possible route that visits each of the given set of cities exactly once and return to the starting city – analogous to minimizing the distance travelled by salesman. The fitness function evaluates by measuring the distance or cost. The fitness function would look like a series of distances plotted, depending on specifics of the problem.

The problem needs sort of a Global Information to find the optimal route and I think, an optimization algorithm that maintains some kind of state or distribution would have an advantage to solve this problem. MIMIC builds Probability Distribution based on fitness and seem to be the ideal candidate for this problem.

Information Classification: GENERAL

Results in Plots:

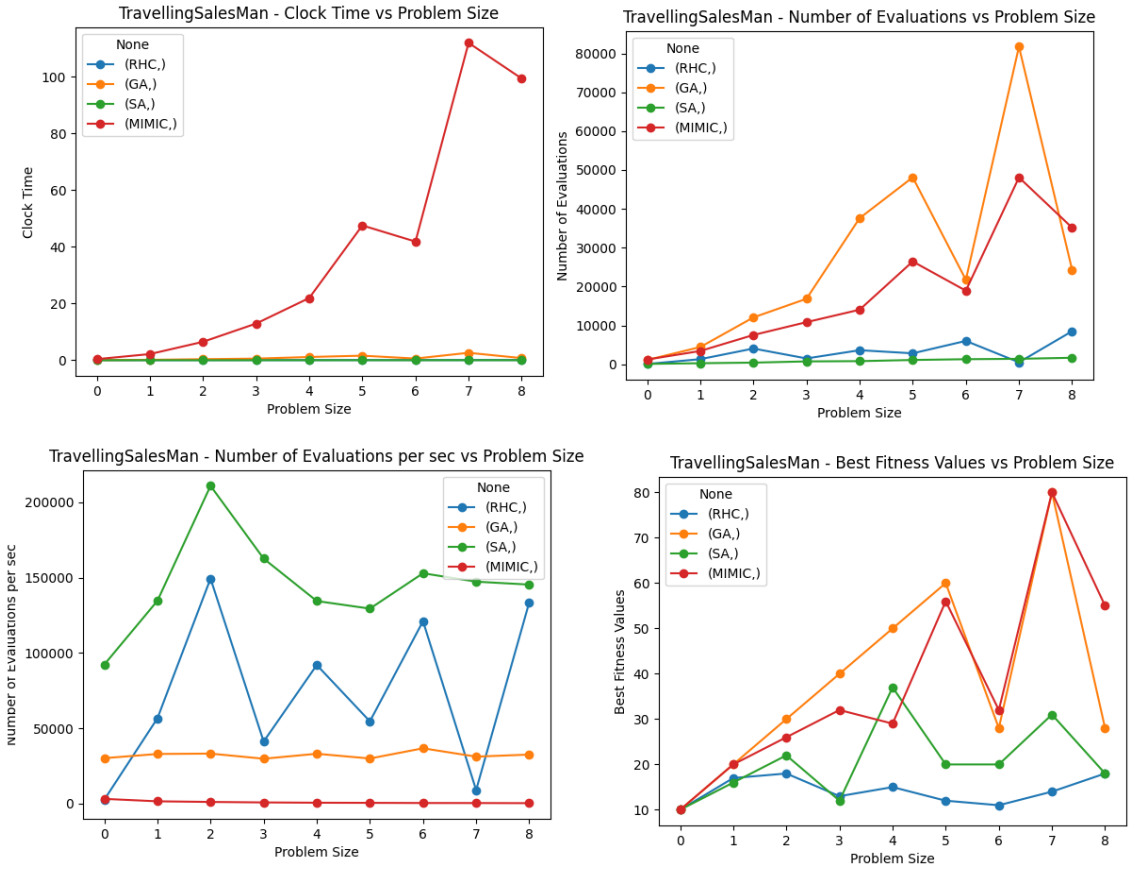

Figure 5: Comparison of Multiple Opt-Algorithms on TravellingSalesMan problem.

MIMIC and Genetic Algorithms seem to outperform other algorithms in achieving high fitness values, however they are expensive as can be seen by the clock time and Number of Evaluations performed per second.

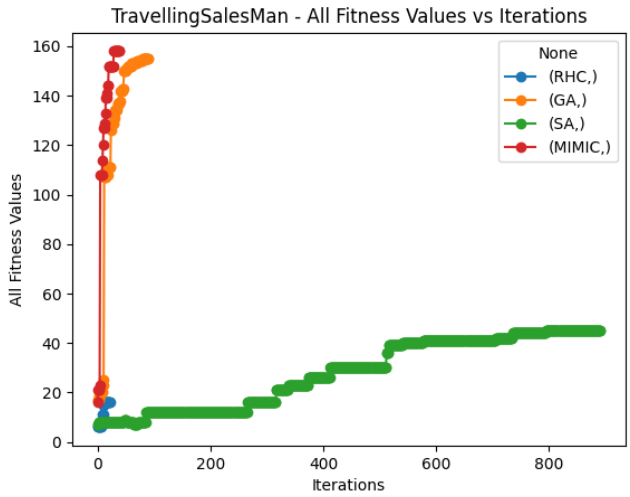

Figure 6: Convergence Plot for TravellingSalesMan Problem

II. Neural Network Optimization

Dataset Description / Recap:

Wine Quality (Link)**:** The features for this Dataset are related to the features of the wine for. E.g. Acidity, residual sugar, density, pH, sulphates, etc. The task is to classify – given the features / attributes of the wine, the quality of wine is either a 'White' wine or 'Red'.

Training: 5170 samples, Test: 1293 samples

Description of Experiment:

Experiment Setup: The Neural Network used in Assignment 1 has been trained by using all Optimization algorithms – Genetic Algorithm, Simulated Annealing, and Randomized Hill Climbing. For comparison, I have replicated the Assignment 1 using mlrose_hiive library for a easier comparison. All the other models will be considered against this Baseline Model. Analysis on other factors like Overfitting, Time Comparison has also been done in the following sections.

Neural Network Results Summary:

	Iterations	Mean Train Accuracy	Mean Test Accuracy	Mean Train F1 Score	Mean Test F1 Score	Mean Train Loss	Mean Test Loss	Mean Clock Time (Training)
Algorithm								
Genetic Algorithm	450	0.898104	0.901469	0.93366	0.935682	3.672688	3.551401	261.4988
Random Hill Climb	450	0.92029	0.922815	0.949648	0.951101	2.873035	2.782024	1.5856
Simulated Annealing	450	0.894217	0.897061	0.934399	0.935994	3.812819	3.710294	2.514676
Gradient Descent (Baseline)	450	0.869246	0.87471	0.920113	0.92315	4.712865	4.51591	0.654117

Result Plots:

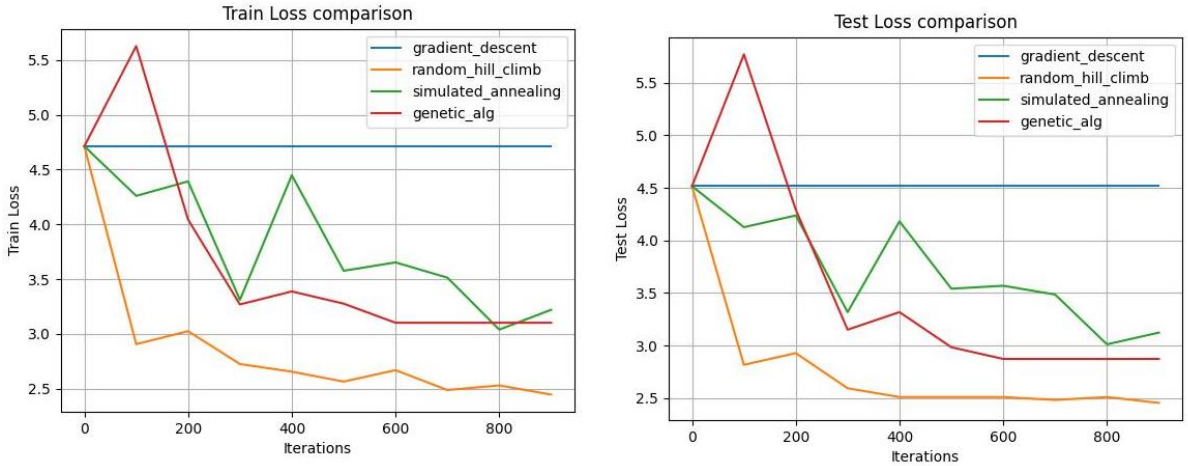

Figure 7: Train and Test Loss Curve for all algorithms with Baseline

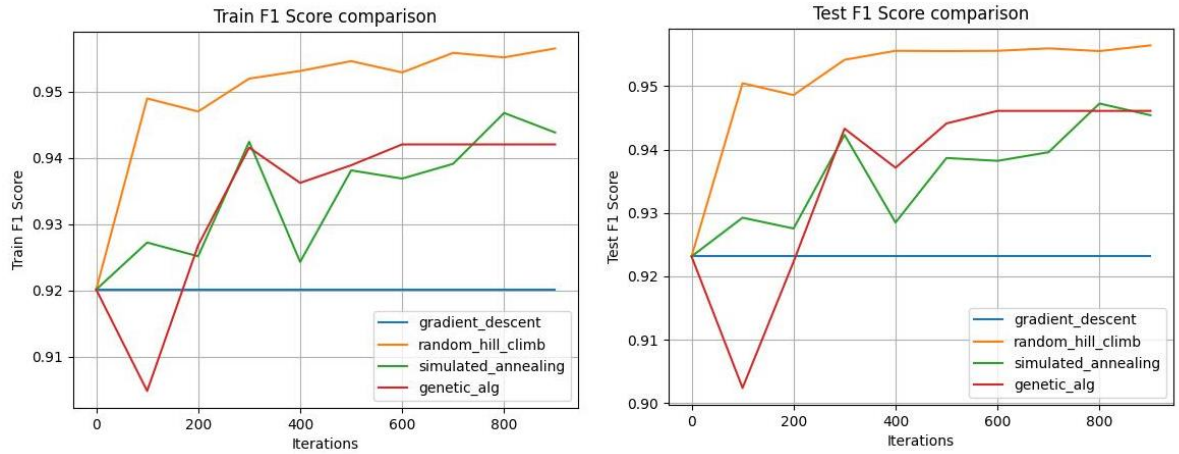

Figure 8: Train and Test F1-score Curve for all algorithms with Baseline

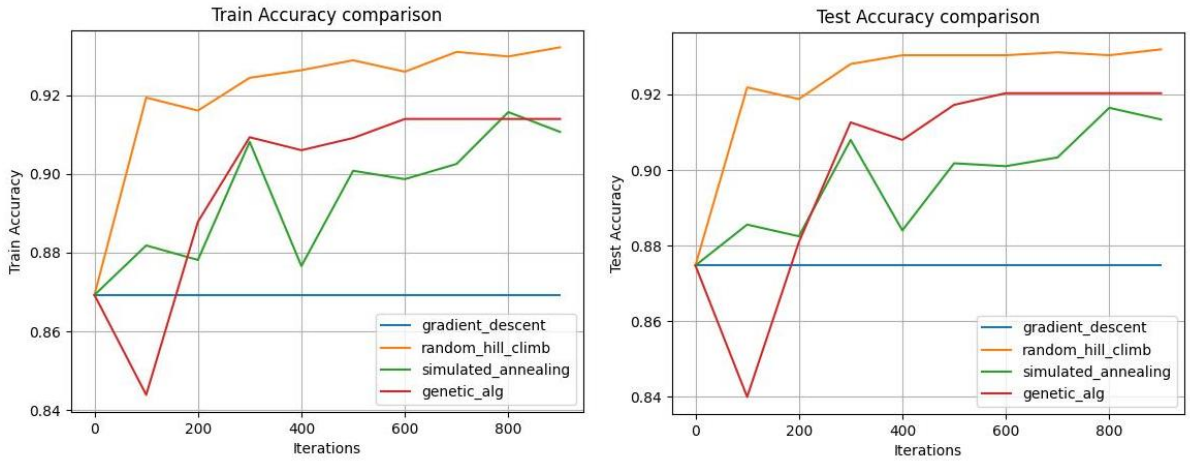

Figure 9: Train and Test Accuracy for all algorithms with Baseline

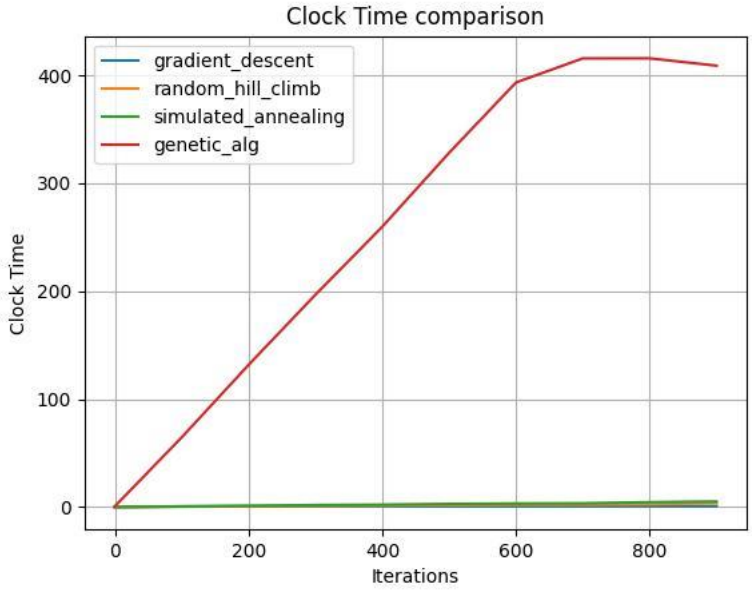

Figure 10: Training Time for all algorithms with Baseline

Comparison and Findings:

Generalization Performance (Accuracy and F1-score)

All the Models generalize well and easily start out a good performance value. This is probably since there is clear separation between the two classes. Comparing the slight differences, RHC outperforms other algorithms by roughly 2%. The optimal metric here would be F1-score due to slight class imbalance rather than accuracy.

Convergence Speed

In terms of Convergence, RHC converges roughly at 300 iterations. Genetic Algorithm takes 100-200 iterations to converge, and Simulated Annealing has still not completely converged even after 900 iterations. However, increasing iterations shows increase in scores which reflects that model learns more as more iterations are included.

Overfitting, Underfitting

The Test and Train performance number are close while looking at the graph. This suggests that there is no kind of Overfitting / Underfitting.

Information Classification: GENERAL

Time to Train

Genetic Algorithms take the most time as we keep on adding iteration. This is since the algorithm produces more and more candidate samples by mutation and cross over. It takes 100x to 200x time as compared to the other models and even more time as compared to the baseline.

Which Algorithm to select (Best Algorithm)

All the algorithms on which we experimented perform equal or better as compared to the Baseline. Based on above parameters, Random Hill Climb seems to be clear winner - this indicates that the Fitness Curve might not be that complex. It is also simpler and memory efficient as compared to other algorithms that might require memory (MIMIC) and simpler compared to Genetic Algorithms. While GA also performs very well, using GA here might be overkill as the Fitness Curve is not that complex.

References:

1. Mlrose_hiive Documentation (Link)
2. Fitness Functions (Link)
3. Overview of Solving TSP using mlrose (Link)
4. Wine Dataset (Link)

Information Classification: GENERAL