Younes Kazemi (40260336) - Evolutionary Computing (2023)

Feature selection with an adaptive multi-objective EA solution

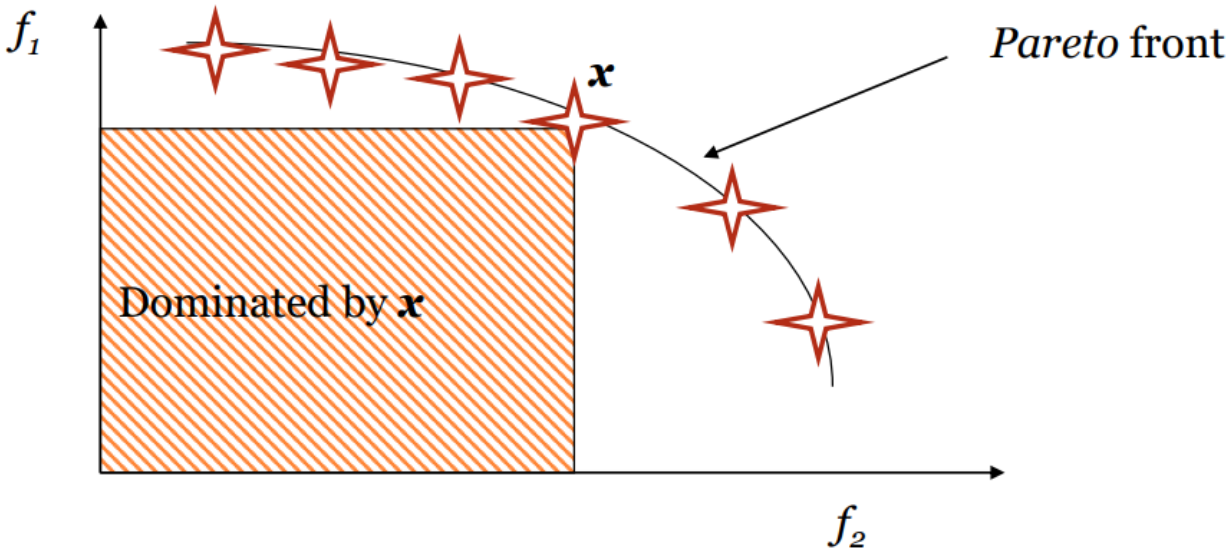


Table of Contents:

Assignment

The provided code implements a Multi-Objective Binary Genetic Algorithm with Adaptive Operator Selection (MOBGA_AOS), specifically designed for multi-objective optimization problems with binary decision variables.

Introduction

Feature selection is crucial in machine learning and data mining. The paper by Xue et al. introduces a multi-objective binary genetic algorithm with adaptive operator selection (MOBGA-AOS) for feature selection, integrating an adaptive mechanism for selecting crossover operators.

Libraries and Data

The implementation utilizes the following libraries: NumPy, Pandas, Matplotlib, Scikit-learn, and Pygmo for hypervolume calculation.

Implementation Details

Framework of MOBGA-AOS

Algorithm 2 *creditAssignment*(P, R, q)

Input: Set of two parents P, set of two children R, the selected crossover operator q
Output: $nReward$, $nPenalty$
$\{P_{nd}, P_d\} \leftarrow dominanceComparison(P)$ // P_{nd} and P_d refer to the sets of non-dominated and dominated solutions, respectively.
if $P_d \neq \emptyset$ **then**
  // One parent dominates the other, and suppose $P_1 \prec P_2$.
  for $i = 1$ *to* 2 **do**
    if $P_1 \prec R_i$ **then**
      $nPenalty_q \leftarrow nPenalty_q + 1$
    else
      $nReward_q \leftarrow nReward_q + 1$
    end if
  end for
else
  // The two parents are non-dominated each other.
  for $i = 1$ *to* 2 **do**
    if $P_1 \not\prec R_i$ && $P_2 \not\prec R_i$ **then**
      $nReward_q \leftarrow nReward_q + 1$
    else
      $nPenalty_q \leftarrow nPenalty_q + 1$
    end if
  end for
end if

Algorithm 1 MOBGA-AOS

Input:
  $maxFEs$: Maximum number of fitness evaluations
  N: Population size
  D: Number of dimensionality (original features)
  M: Number of objectives
  Q: Number of crossover operators
  LP: Number of repeated generations for OSP
Output:
  Optimal feature subsets
$P \leftarrow initPop(N)$
Initialize $nReward$, $nPenalty$, $RD_{LP\times Q}$, $PN_{LP\times Q}$ according to Eqs. (4)–(5)
$\vec{P} = \{p_1, p_2, ..., p_Q\} \leftarrow initOSP(Q)$
$k \leftarrow 0$
$nFE \leftarrow 0$
$P_{new} \leftarrow \emptyset$
while $nFE < maxFEs$ **do**
  for $i = 1$ to $N/2$ **do**
    $operator\_idx \leftarrow rouletteWheelSelection(\vec{P})$
    Randomly select two individuals as parents: P_p
    $P_c \leftarrow crossover(P_p, operator\_idx)$
    $P_c \leftarrow uniformMutation(P_c)$
    $nFE \leftarrow nFE + 2$
    $\{nReward, nPenalty\} \leftarrow creditAssignment(P_p, P_c)$
    Add P_c to P_{new}
  end for
  $k \leftarrow k + 1$
  Append $nReward$ to the k^{th} row of $RD_{LP\times Q}$
  Append $nPenalty$ to the k^{th} row of $PN_{LP\times Q}$
  if $k = LP$ **then**
    $\vec{P} \leftarrow updateOSP(RD_{LP\times Q}, PN_{LP\times Q})$
    $k = 0$
  end if
  $R \leftarrow P \cup P_{new}$
  $P \leftarrow environmentalSelection(R)$
  Select non-dominated solutions in P as PF
end while
Optimal feature subsets $\leftarrow PF$
return Optimal feature subsets

Operations

Genetic algorithms, Selection, Crossover, Mutation,

1. Population initialization

straightforward method to represent the feature selection problem, i.e., 0–1 encoding schema. To be more specific, if the encoding of an individual is a vector of 0101100, it means that the number of the original features is seven, and the second feature, the fourth feature, and the fifth feature is selected. After encoding, the initial population can be generated by a random approach.

2. Population Representation:

The population is represented as binary strings, and operators such as one-point, two-point, uniform, shuffle, and reduced surrogate crossovers are used for recombination.

3. Fitness Calculation:

The purpose of the multi-objective feature selection problem is to find a set of optimal features with high classification accuracy and small solution size (numb of features)

k nearest neighbors (k-NN, k = 3) is used as the classifier to evaluate solutions, and n fold cross validation (n = 3) is used for the k-NN.

$$minf_1(X) = \left(\frac{1}{n}\sum_{l=1}^{n}\frac{N_{Error}}{N_{All}}\right) \times 100\% \qquad minf_2(X) = \sum_{i=1}^{D} x_i$$

4. Parent Selection:

Randomly select two individuals as parents:

5. Crossover operator pool

Many different binary genetic crossover operators have been designed. Five of them that have unique search abilities are elected for the generation of new solutions in MOBGA-AOS.

Single-point, Two-Point, Uniform, Shuffle and Reduce surrogate

6. Credit assignment

We suppose there are Q > 1 operators in the operator pool, and the qth operator is selected for reproduction. Our intention lies in rewarding operators that can produce promising children that are comparable to their parents. To attain this, two vectors, i.e., nReward and nPenalty, are constructed to record the evolution information of the used operators in each generation. In GAs, two parents can generate two children through the reproduction process. Hence, the Pareto dominance relationship between the parents and the children is analyzed to update nReward and nPenalty. According to the Pareto dominance relationship between the two parents, there are two cases:

6.1 One parent dominates the other

Let us suppose that parent i is dominated by parent j, each child is used to compare the Pareto dominance relationship with parent j, respectively.

If the child is not dominated by parent j, then nRewardq + 1, otherwise nPenaltyq + 1.

6.2 The two parents are non-dominated each other

Each child is used to compare the Pareto dominance relationship with the two parents, respectively. If the two parents do not dominate the child simultaneously, nRewardq +1, otherwise nPenaltyq + 1.

7. Mutation:

Mutation Uniform mutation with rate of Pm

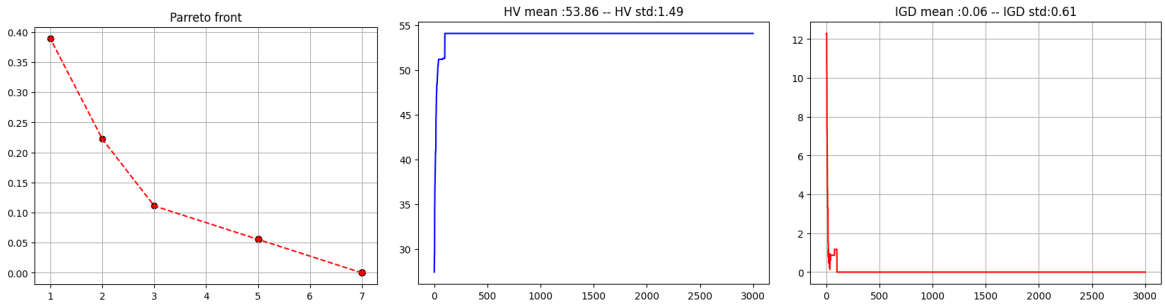
8. Environmental Selection

Fast nondominated sorting and crowding distance to select survivors.

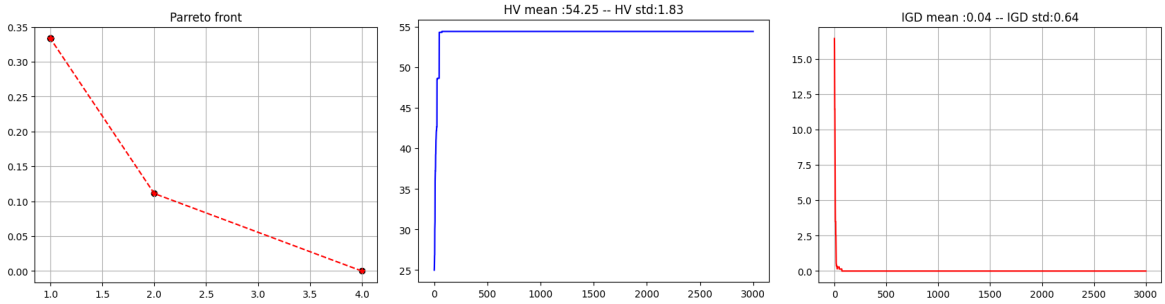
Results

1. DS02:

Train set:

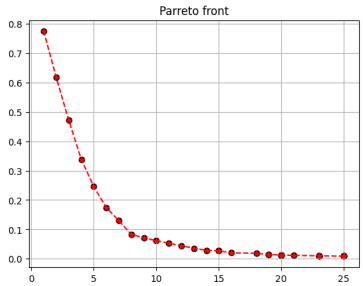


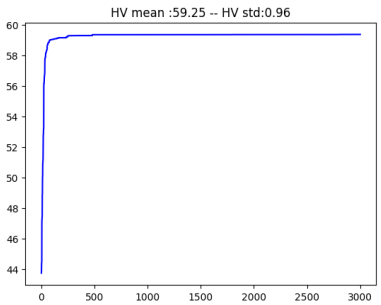
Test set:

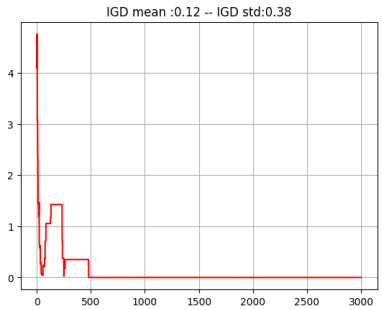


2. DS04:

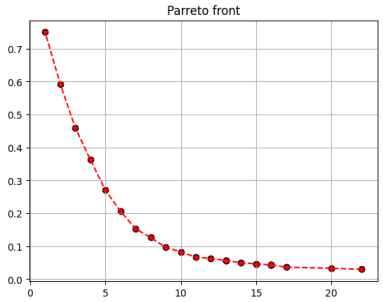
Train set:

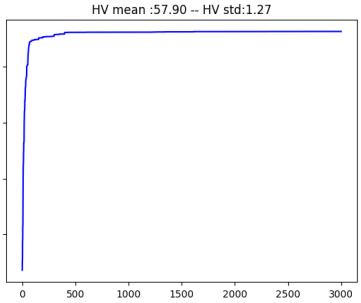


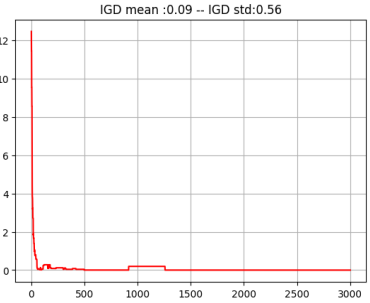




Test set:

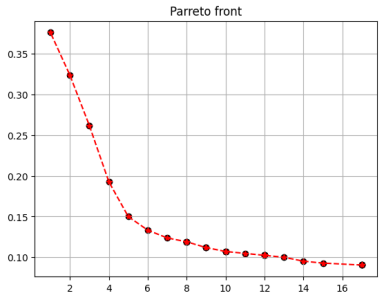


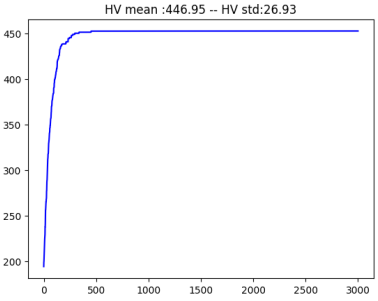


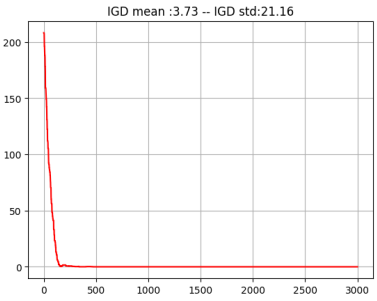


3. DS05:

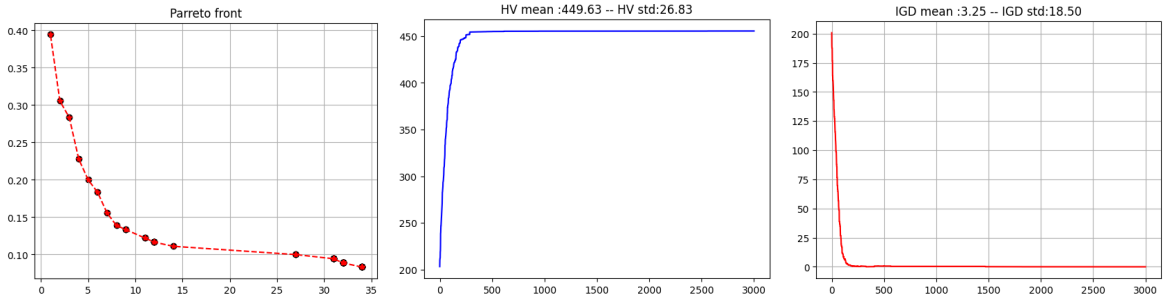
Train set:





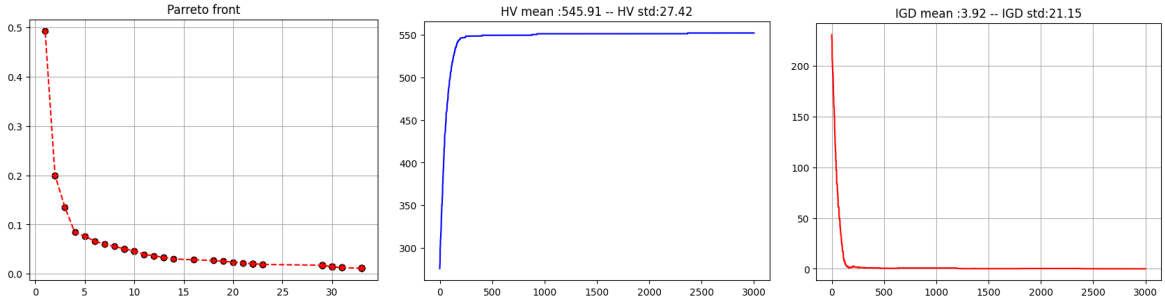


Test set:

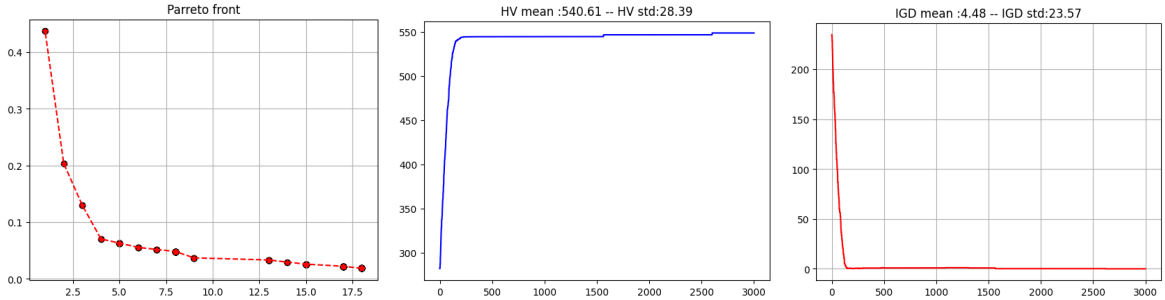


4. DS07:

Train set:

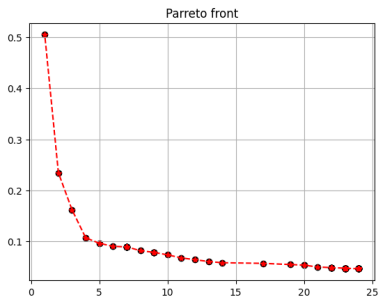


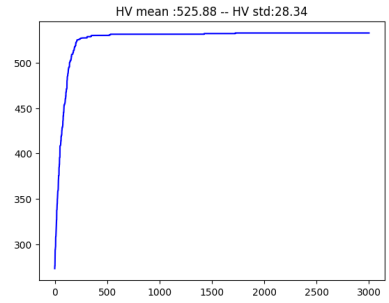
Test set:

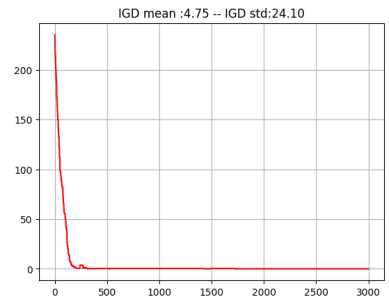


5. DS08:

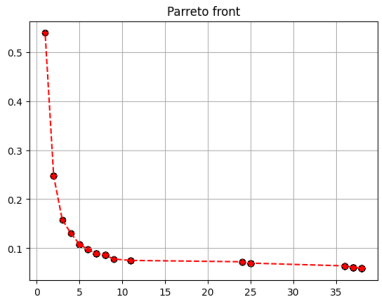
Train set:

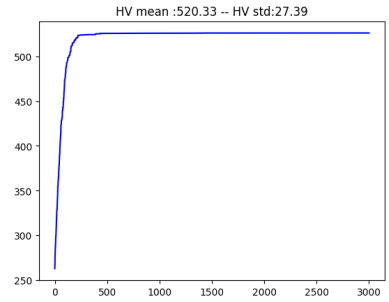


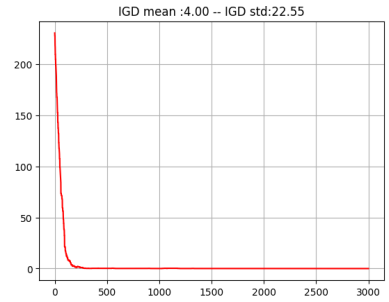




Test set:

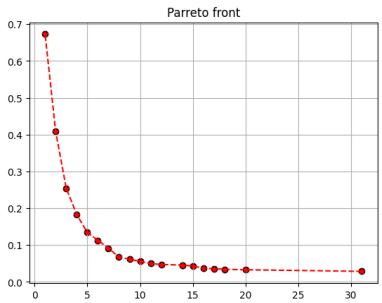


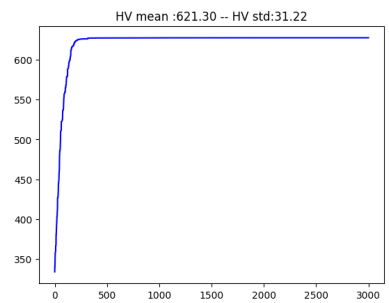


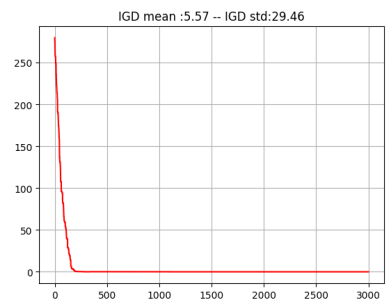


6. DS10:

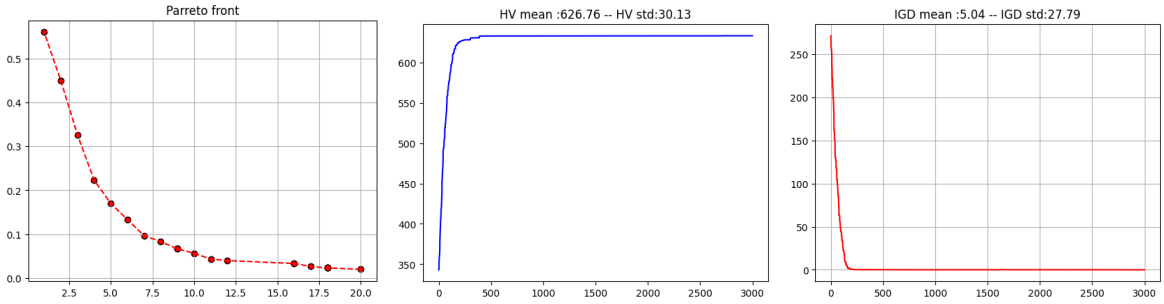
Train set:







Test set:



Task:

1. Implement MOBGA-AOS : **Done! (implemented well with Python OOP)**
2. Plot the obtained Pareto front of your implementation. : **Done! (you can see all plots in this report)**
3. Suggest a solution to converge to optimal solutions more quickly for datasets with a larger number of features (more than 250 features). Justify your solution. **Done!**
    a. I remove duplicate individuals in my population so the chance of crossover between different individuals goes up and makes converge faster
    b. also I cached fitness calculation that makes more faster and efficient algorithm so we do not need to recalculate and fit to KNN the same individual
4. Plot the hypervolume value for each generation. **Done! (you can see all plots in this report)**
5. Run your algorithm multiple times. **Done! (I ran multiple times for each data set both for training and testing)**