

On initial population generation in feature subset selection

DENIZ, A and KIZILOZ, HE

Available from Sheffield Hallam University Research Archive (SHURA) at:

<http://shura.shu.ac.uk/30147/>

This document is the author deposited version. You are advised to consult the publisher's version if you wish to cite from it.

Published version

DENIZ, A and KIZILOZ, HE (2019). On initial population generation in feature subset selection. *Expert Systems with Applications*, 137, 11-21.

Copyright and re-use policy

See <http://shura.shu.ac.uk/information.html>

On initial population generation in feature subset selection

Ayça Deniz^a, Hakan Ezgi Kiziloz^{b,*}

^a*Middle East Technical University*

^b*University of Turkish Aeronautical Association*

Abstract

Performance of evolutionary algorithms depends on many factors such as population size, number of generations, crossover or mutation probability, etc. Generating the initial population is one of the important steps in evolutionary algorithms. A poor initial population may unnecessarily increase the number of searches or it may cause the algorithm to converge at local optima. In this study, we aim to find a promising method for generating the initial population, in the Feature Subset Selection (FSS) domain. FSS is not considered as an expert system by itself, yet it constitutes a significant step in many expert systems. It eliminates redundancy in data, which decreases training time and improves solution quality. To achieve our goal, we compare a total of five different initial population generation methods; Information Gain Ranking (IGR), greedy approach and three types of random approaches. We evaluate these methods using a specialized Teaching Learning Based Optimization searching algorithm (MTLBO-MD), and three supervised learning classifiers: Logistic Regression, Support Vector Machines, and Extreme Learning Machine. In our experiments, we employ 12 publicly available datasets, mostly obtained from the well-known UCI Machine Learning Repository. According to their feature sizes and instance counts, we manually classify these datasets as small, medium, or large-sized. Experimental results indicate that all tested methods achieve similar solutions on small-sized datasets. For medium-sized and large-sized datasets, however, the IGR method provides a better starting point in terms of execution time and learning performance. Finally, when compared with other studies in literature, the IGR method proves to be a viable option for initial population generation.

Keywords: Feature subset selection, Initial population, Multiobjective optimization

1. Introduction and Background

2 With advances in technology and the effects of globalization, electronic devices get
3 cheaper and cheaper. As of today, technology has spread among users having various types
4 of demographics and it has been an indispensable part of our lives. As a result, 2.5 quintil-
5 lion bytes of data had been created every day in 2012 and this volume increases massively

*Corresponding author

Email addresses: ayca.deniz@metu.edu.tr (Ayça Deniz), hakanezgi@etu.edu.tr (Hakan Ezgi Kiziloz)

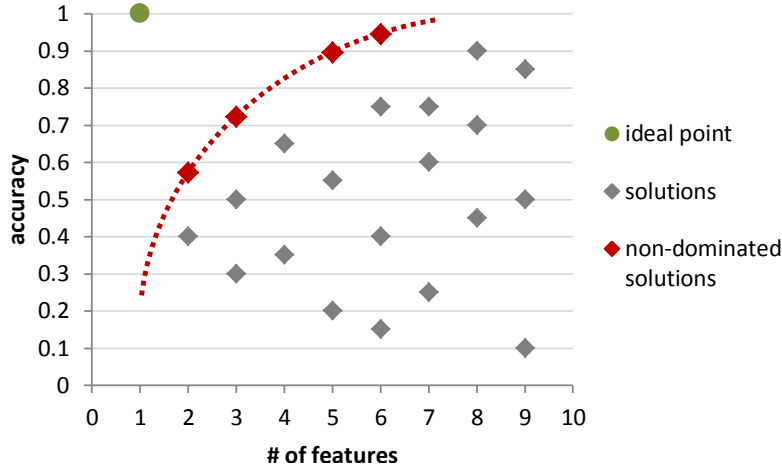


Figure 1: Non-dominated solutions fitting to a pareto curve in a multiobjective FSS problem (Deniz et al., 2017).

6 every passing minute (BBC, March 2014). It is not feasible to analyze this amount of data
7 by manual processing, hence, we need to utilize machine learning techniques to do it au-
8 tomatically. However, building an efficient and accurate machine learning models is not a
9 straightforward process. Also, training a machine learning model requires analyzing every
10 single feature of every single instance. Therefore, removing redundant and/or irrelevant fea-
11 tures from data could be beneficial in terms of computation time and learning performance.
12 Feature Subset Selection (FSS) has been a preferable tool to achieve this goal (Guyon &
13 Elisseeff, 2003).

14 FSS is the process that tries to remove redundant and/or irrelevant features from data.
15 Since these features may not be clear initially, the utilization of such searching algorithms for
16 finding the features that represent the data best is crucial. In this study, we use the MTLBO-
17 MD searching algorithm to find the best feature subset. MTLBO-MD is a multiobjective
18 teaching learning based optimization algorithm, proposed in our previous study (Kiziloz
19 et al., 2018). Detailed information on the MTLBO-MD algorithm is given in subsection 2.1.
20 We need a multiobjective search algorithm since there are two objectives in this problem
21 domain, namely, decreasing number of features and increasing classification performance
22 (accuracy). In a multiobjective optimization environment, different solutions may dominate
23 each other in different objectives, hence, there may exist a solution set rather than only
24 one solution (Horn et al., 1994). Figure 1 shows a sample of non-dominated solutions in a
25 hypothetical problem.

26 Evolutionary based algorithms are utilized in many combinatorial and optimization prob-
27 lems and they require an initial population to begin their search. Traditionally, this initial
28 population is set randomly (Michalewicz, 1996). However, poor selection of initial popu-
29 lation may lead to getting stuck in local optima rather than finding the global optimum
30 (Toğan & Daloglu, 2008). Also, choosing the initial population wisely could possibly lead

31 to faster convergence to the global optimum. For this reason, we investigate the effects of
32 using different initial population generation methods in FSS domain. Our main motivation
33 is to analyze empirical results and find a promising initial population generation method in
34 this domain.

35 There are many studies proposing different strategies to generate an initial population
36 in evolutionary algorithms. Maaranen et al. (2007) show that the initial population may
37 have a considerable effect on the objective function in continuous optimization problems.
38 They test different initial population seeds and conclude that smarter initial populations
39 may decrease computation time or may bring out better solutions. Similarly, Karci (2004)
40 proposes a method to initialize the population for continuous global optimization problems.
41 The study emphasizes that the initial population should be evenly distributed for not getting
42 stuck in local optima. Three benchmark problems having many dimensions were used in
43 their experiments and it is seen that their method finds near-optimal solutions. Besides,
44 Diaz-Gomez & Hougen (2007) propose five different (two gene-level, two chromosome-level,
45 and one population-level) metrics to evaluate population diversity. However, they do not
46 compare these metrics with empirical results.

47 Studies investigating the initial population for well-known optimization problems also ex-
48 ist. Hill (1999) proposes a heuristic for generating the initial population for two-dimensional
49 knapsack problems, which performs better than the traditional methods according to their
50 experiments. Similarly, Escobar et al. (2011) propose a constructive heuristic algorithm to
51 generate the initial population for transmission expansion planning problem. They use real
52 data in their experiments and their algorithm gives better results than random initializa-
53 tion. In addition, Jorapur et al. (2016) proposes a population initialization algorithm for
54 the job shop scheduling problem. Their algorithm uses random job based initialization after
55 a simple job based representation and obtains better results than mere random based one.
56 Moreover, Deng et al. (2015) propose a k -means based initial population generation algo-
57 rithm for the travelling salesman problem. Experimental results indicate high performance
58 improvement as compared to generating an initial population in a random or greedy fash-
59 ion. Meanwhile, setting the initial population in k -means clustering is also very important
60 since it may affect the clustering result dramatically. Zhou et al. (2018) propose a method,
61 called SeedClust, which generates the best chromosome for automatic k -means clustering
62 by applying an adaptive genetic algorithm. Experiment results show that SeedClust is an
63 effective method for finding an effective initial population.

64 Up to our best knowledge, the effects of the initial population have not been investigated
65 thoroughly in the FSS domain. On the other hand, all mentioned studies state, with empir-
66 ical results on different problem domains, that initial population should be selected wisely,
67 rather than being generated randomly. Wise selection methods refer to statistical or greedy
68 approaches, both of which are included in our tests.

69 The main contribution of this study is to investigate the effects of using different initial
70 population generation methods in FSS domain. For this reason, we compare five different
71 methods (traditional methods along with our proposed methods) using various datasets and
72 analyze their effectiveness with empirical results. FSS is generally not the main functionality
73 of an expert system, yet, applying it to remove redundant data becomes a crucial step as



Figure 2: Chromosome structure (1 indicates selected features, 0 otherwise).

74 dataset sizes grow larger. Moreover, it should be applied without compromising efficiency.
 75 We expect our study to help researchers apply FSS with higher accuracy using less resources
 76 when building an expert system.

77 The rest of the manuscript is organized as follows. In Section 2, our model is introduced
 78 with the following subsections: population selection, initial population generation methods,
 79 and applied machine learning techniques. Experimental results and discussion are given
 80 in Section 3. Finally, concluding remarks and possible future works are shared in the last
 81 section, Section 4.

82 2. Model

83 In this section, we introduce the model used in our study. First, we describe the pop-
 84 ulation selection algorithm, MTLBO-MD, in detail. Then, we give the initial population
 85 generation methods used. Finally, we briefly mention the machine learning techniques used
 86 for evaluating individuals.

87 2.1. Individual description and population selection

88 Finding the best population in a multiobjective environment is not a straightforward
 89 process since an improvement in one objective may worsen other objectives. In a previous
 90 study (Kiziloz et al., 2018), we proposed a promising algorithm, called Multiobjective Teach-
 91 ing Learning Based Optimization with Minimum Distance (MTLBO-MD), to overcome this
 92 problem. For FSS domain, we define a chromosome as given in Figure 2. Here, a chromo-
 93 some consists of 1s and 0s, indicating whether its corresponding feature in the dataset is
 94 selected or not, respectively.

95 In a multiobjective domain, an individual in the population dominates another one if
 96 and only if it gives better result in at least one of the objectives while all other objectives
 97 give the same result. If two individuals can not dominate each other, then they are called
 98 non-dominated to each other. MTLBO-MD consists of two phases: the teacher phase and
 99 learner phase (see Figure 3). In teacher phase, all non-dominated individuals are found and
 100 the one that is closest to the ideal point is chosen as the teacher while the other individuals
 101 are set as students. The ideal point in feature subset selection domain is the point where the
 102 *number of features* = 1 and *accuracy* = 1.0, i.e. being able to classify perfectly using only
 103 one feature. After the teacher is selected, a crossover is applied between the teacher and
 104 all students, separately. We keep both old and new individuals (students) in the possible
 105 population set. Teacher phase of the algorithm ends with this operation and the learner
 106 phase begins. In the learner phase, a crossover is applied to two random students for

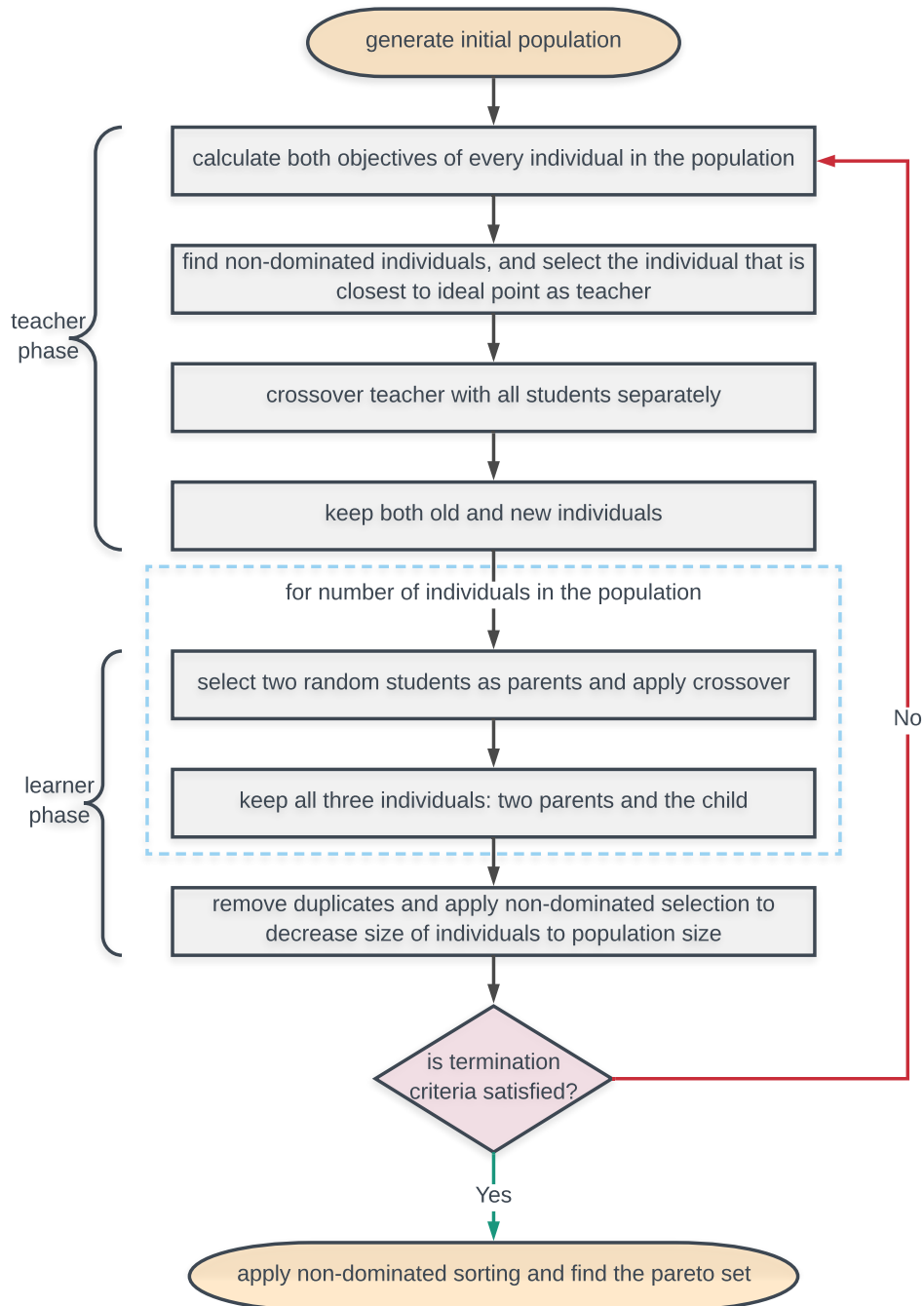


Figure 3: MTLBO with Minimum Distance (MTLBO-MD).

107 the number of population size. Meanwhile, all new individuals are added to the possible
108 population set. When the crossover is complete, first, duplicate individuals in the possible
109 population set are removed and the number of individuals is decreased to population size
110 via non-dominated sorting. For this purpose, individuals are divided into fronts (i.e. the
111 first front includes all non-dominated solutions, the second front includes solutions that are
112 only dominated by first front solutions and are non-dominated to each other, etc.). The
113 population is filled by selecting from these fronts in order. Crowding distance method is
114 used to select individuals constituting a front if only a part of that front is needed to fill the
115 population. When the teacher and learner phase run for the specified number of generations,
116 the algorithm terminates after finding the pareto set.

117 We used crossover and mutation operators in the generation of new chromosomes. Half
118 uniform crossover is implemented for the crossover part, and bit-flip mutation is imple-
119 mented for the mutation part. In half uniform crossover, one new chromosome, namely
120 child chromosome, is generated by using two existing chromosomes, namely parent chromo-
121 somes. Feature genes of the parent chromosomes are compared, and the same ones are kept
122 in the child chromosome. If a feature gene is different in the parents, then one of them is
123 randomly chosen and transferred to the child chromosome. Mutation, on the other hand,
124 operates on a single chromosome. It inverts one of the feature genes randomly and generates
125 a new chromosome. We set the mutation probability to 0.02 in our implementation.

126 2.2. Initial population generation methods

127 We initially designed four different initial population generation methods, as described
128 below. The fifth method, SR, is introduced as a post hoc study after obtaining initial test
129 results.

130 *Random (Rnd)*: In this method, initial population is generated in a complete random
131 fashion.

132 *Distinct Random (DR)*: Rather than complete randomness, we wanted to add a heuristic
133 into random population generation and investigate its effects. For this purpose, we randomly
134 generated individuals that is 10 times the population size and selected the most distinct
135 individuals among them as the initial population. Here, we use the Hamming Distance, i.e.
136 the number of different features, as distinctness metric between two chromosomes.

137 *Information Gain Ranking (IGR)*: In this method, we utilized a filter based feature
138 selection approach, information gain (Quinlan, 1986), to find the most promising features in
139 our dataset. Then we generate the initial population as follows: the first individual is the
140 chromosome having only the most promising feature selected, the second individual is the
141 chromosome having the most two promising features selected, and so on. We continue to
142 generate individuals in this fashion until population size is met.

143 *Greedy (Gr)*: Finally, we utilized a greedy approach, sequential forward selection, to
144 generate the initial population. For this purpose, we evaluate all chromosomes having only
145 one feature selected, and pick the feature with the highest accuracy value. We add the
146 individual having this picked feature selected into our initial population. Then, for the
147 second individual, we fix the picked feature as selected and evaluate all possible chromosomes
148 having two features selected. Now, we pick the features achieving the highest accuracy value,

149 and add this individual into the initial population. Search for the third individual is similar,
150 picked two features are fixed as selected, and all chromosomes having three individuals are
151 evaluated. This iteration continues until population size is met. The main difference of
152 this method from others is its requirement to evaluate many chromosomes for generating
153 the initial population. This method evaluates many possibilities and could be considered
154 as an extreme case, yet, its solutions are valuable for evaluating the performances of other
155 methods.

156 *Small Random (SR)*: In Rnd and DR, the initial population is deployed in a totally
157 random fashion, giving it a small chance to find the global optima in small population size
158 and number of generations. Accordingly, we introduced the SR method as a smarter option
159 for random based initial population generation methods. In this method, the first individual
160 is the chromosome having only one randomly selected feature. This individual is added to
161 the initial population. The second individual is the chromosome having two features: the
162 first one is the selected feature for the first individual, the second one is randomly selected.
163 After adding this individual into the initial population, the third individual is generated
164 similarly: the first two features selected from the second individual, the third feature is
165 selected randomly. Iteration continues until population size is met.

166 2.3. Applied machine learning techniques

167 We used Logistic Regression (LR), Support Vector Machines (SVM) and Extreme Learn-
168 ing Machine (ELM) to evaluate solution sets obtained by MTLBO-MD algorithm. LR uses
169 a probabilistic model to perform classification, SVM tries to maximize the distance between
170 different classes, and ELM utilizes the power of neural networks.

171 *Logistic Regression*: LR analyzes the similarity of the data points to estimate the occur-
172 rence probability of an event. Once this probability is calculated, it decides on the occurrence
173 result and performs classification accordingly. We used a Matlab function, *glmfit*, for LR
174 classification.

175 *Support Vector Machines*: SVM constructs a line between different classes by maximiz-
176 ing the distance between the line and closest data points to the line. We used a Matlab
177 function, *fitcsvm*, for SVM classification. We chose kernel type as linear and set normaliza-
178 tion parameter to *True*. Other than these, default parameters of the built-in function are
179 used.

180 *Extreme Learning Machine*: ELM is a feedforward neural network consisting of a single
181 hidden layer. ELM does not need tuning which distinguishes it from other neural networks.
182 Consequently, the training time of ELM is less than other traditional neural networks. We
183 used an ELM library, developed by Huang et al. (2011), for ELM classification. We set the
184 activation function as sigmoidal and node count to 20 in the hidden layer.

185 3. Experimental Results

186 In this section, we describe our experimental environment, problem instances, and ob-
187 tained results. We carried out the experiments on a computer with AMD Ryzen 7 1800X

Table 1: Specification of the datasets used in the experiments.

Dataset	Problem ID	Number of features	Actual number of classes	Number of instances	Class proportions
Coverttype	CT	54	7	581,012	43% - 57%
Mushrooms	MR	22	2	8124	48% - 52%
Spambase	SB	57	2	4601	39% - 61%
Nursery	NU	8	5	12,960	50% - 50%
Connect-4 Opening	C4	42	3	67,557	27% - 73%
Waveform	WF	40	3	5000	50% - 50%
Financial	FI	93	2	17,108	3% - 97%
Pima Indian Diabetes	PM	8	2	768	35% - 65%
Breast Cancer	BC	9	2	699	35% - 65%
Ionosphere	IO	34	2	351	36% - 64%
Wisconsin Breast Cancer	WBC	30	2	569	37% - 63%
Musk (Version 2)	MU	168	2	6598	15% - 85%

188 Eight-Core Processor with a 3.6 GHz clock rate and 16 GB of main memory. We used
 189 MATLAB 2015a for evaluating individual accuracy, and Java for remaining parts.

190 In our study, we used 12 datasets, 11 of which are obtained from the University of
 191 California UCI Machine Learning Repository (Dua & Graff, 2017). One remaining dataset,
 192 *Financial*, is retrieved from a study by Pacheco et al. (2009). Detailed information on
 193 datasets is given in Table 1. Most of the datasets used in this study have two actual classes.
 194 In different cases, datasets are reduced to two classes with the selection of the classes having
 195 most instances. Categorical values in datasets are converted to numerical values by arbitrary
 196 coding of symbols, in the preprocessing phase of the study.

197 In supervised learning, classification accuracy mainly relies on how well training data
 198 represents actual data. To eliminate any bias towards one class, the data is generally split
 199 into k -partitions and each partition is used for testing in separate turns, while all remaining
 200 partitions are used for training. This methodology is called k -fold cross-validation. However,
 201 some of our datasets have a high amount of instances (the number of instances increases
 202 up to 581,012) and application of k -fold cross-validation on these datasets would consume
 203 an enormous amount of time. For this reason, we first reduced the instance sizes of the
 204 CT, NU, WF, and FI datasets to 10,000, with respect to their original class proportions,
 205 and then applied 5-fold cross-validation on all datasets. A validation set is not used in the
 206 experiments.

207 In an evolutionary algorithm, especially in MTLBO-MD, population size and number of
 208 generations are two main factors that affect the quality of the results. In our previous study
 209 (Deniz et al., 2017), we discovered with experimental results that, selecting population size
 210 as 40 and the number of generations as 60 could lead to finding good enough solutions in
 211 an acceptable amount of time. However, we opted to halve both values since we are only
 212 analyzing the effects of initial population generation methods in this study. This choice
 213 would limit the exploration and exploitation capability of the MTLBO-MD algorithm, and
 214 hence, initial population generation would become more important. As a result, population
 215 size and the number of generations are selected as 20 and 30, respectively.

Table 2: Maximum accuracy, number of features at maximum accuracy, and execution time values for each initial population generation method and machine learning technique.

Dataset (Feature size)	Method	Max. accuracy			# of features at max. accuracy			Exec. time (sec.)		
		LR	SVM	ELM	LR	SVM	ELM	LR	SVM	ELM
CT (54)	Rnd	0.775	0.778	0.685	10	14	8	1341.9	20170.5	251.6
	Gr	0.779	0.776	0.666	10	7	2	927.5	16932.8	385.8
	IGR	0.772	0.773	0.655	8	6	3	86.6	6727.1	59.3
	DR	0.777	0.776	0.586	11	11	10	590.5	13769.8	135.0
	SR	0.751	0.621	0.673	3	3	4	79.3	592.7	35.7
MR (22)	Rnd	0.938	0.958	0.981	3	8	4	113.0	3621.5	83.5
	Gr	0.955	0.952	0.990	9	6	2	294.0	2875.4	113.7
	IGR	0.950	0.957	0.989	6	8	3	44.6	5703.3	18.4
	DR	0.945	0.950	0.990	8	6	4	87.4	3758.7	51.5
	SR	0.938	0.956	0.947	4	6	4	23.8	4481.5	20.7
SB (57)	Rnd	0.917	0.919	0.882	15	15	14	138.0	4561.8	135.5
	Gr	0.922	0.922	0.889	14	15	8	146.8	2748.6	220.2
	IGR	0.900	0.906	0.882	8	13	7	28.9	978.5	26.3
	DR	0.915	0.910	0.887	18	25	12	121.8	3719.0	63.8
	SR	0.869	0.854	0.818	13	13	8	34.1	2410.0	33.3
NU (8)	Rnd	1.000	1.000	1.000	1	1	1	34.4	168.4	31.0
	Gr	1.000	1.000	1.000	1	1	1	40.4	104.5	30.9
	IGR	1.000	1.000	1.000	1	1	1	20.0	43.0	8.7
	DR	1.000	1.000	1.000	1	1	1	29.0	443.5	27.4
	SR	1.000	1.000	1.000	2	2	2	13.9	210.0	10.8
C4 (42)	Rnd	0.805	0.797	0.792	21	11	9	308.3	15743.8	225.4
	Gr	0.818	0.728	0.798	16	1	14	188.1	4872.9	522.6
	IGR	0.806	0.813	0.793	13	17	11	46.2	10347.2	124.8
	DR	0.813	0.803	0.791	17	16	11	156.5	12621.0	160.6
	SR	0.744	0.738	0.759	11	5	7	74.6	1206.8	86.2
WF (40)	Rnd	0.923	0.924	0.904	9	14	7	47.4	2314.4	74.7
	Gr	0.924	0.926	0.902	9	9	7	47.5	2392.2	125.7
	IGR	0.925	0.929	0.903	10	11	6	18.4	790.2	11.6
	DR	0.922	0.926	0.906	13	14	6	24.0	2283.7	33.7
	SR	0.900	0.893	0.871	9	6	4	15.3	497.4	12.2
FI (93)	Rnd	0.966	0.966	0.966	8	11	15	2566.8	71212.4	203.2
	Gr	0.966	0.966	0.966	1	1	1	547.5	23589.3	672.0
	IGR	0.966	0.966	0.966	1	1	1	71.2	302.8	28.2
	DR	0.966	0.966	0.966	14	15	10	2209.0	76677.8	119.8
	SR	0.966	0.966	0.966	1	1	1	45.1	615.9	30.9
PM (8)	Rnd	0.773	0.776	0.727	5	6	1	2.9	23.0	7.1
	Gr	0.777	0.776	0.741	6	6	1	2.4	23.7	5.5
	IGR	0.773	0.776	0.746	5	6	1	4.6	12.3	3.5
	DR	0.777	0.773	0.720	6	3	1	2.9	19.7	6.3
	SR	0.771	0.773	0.714	4	3	4	3.5	7.9	2.7
BC (9)	Rnd	0.969	0.971	0.966	4	5	4	2.6	12.4	6.8
	Gr	0.967	0.971	0.966	5	5	3	3.6	11.6	6.1
	IGR	0.967	0.971	0.970	5	5	3	1.8	4.1	3.2
	DR	0.969	0.971	0.967	3	5	3	3.3	8.7	5.9
	SR	0.966	0.969	0.969	4	6	4	1.2	6.2	2.4
IO (34)	Rnd	0.900	0.906	0.917	6	12	5	40.7	89.8	26.6
	Gr	0.906	0.912	0.917	8	7	3	114.0	77.6	35.2
	IGR	0.858	0.877	0.923	6	5	3	4.3	28.8	6.9
	DR	0.895	0.915	0.926	10	12	4	20.0	53.5	12.1

Continued on next page

Table 2 – continued from previous page

Dataset (Feature size)	Method	Max. accuracy			# of features at max. accuracy			Exec. time (sec.)		
		LR	SVM	ELM	LR	SVM	ELM	LR	SVM	ELM
	SR	0.895	0.877	0.915	7	9	3	14.2	33.8	5.5
WBC (30)	Rnd	0.979	0.981	0.954	7	10	7	17.0	74.4	32.9
	Gr	0.981	0.984	0.930	6	9	3	15.7	76.5	27.9
	IGR	0.979	0.975	0.928	6	5	3	4.6	14.9	7.2
	DR	0.974	0.979	0.960	5	7	5	13.1	32.3	21.7
	SR	0.979	0.984	0.947	6	9	5	6.6	33.6	7.3
MU (168)	Rnd	0.938	0.944	0.866	72	69	37	880.8	55579.0	293.3
	Gr	0.910	0.921	0.890	11	10	9	354.8	20854.9	978.8
	IGR	0.920	0.920	0.880	10	16	5	25.6	3306.1	40.9
	DR	0.941	0.943	0.867	64	50	30	628.6	43691.9	180.7
	SR	0.868	0.896	0.870	6	7	5	19.1	1687.3	39.4

219

216 Accuracy results of the non-dominated solutions, execution times and the number of
 217 features for each initial generation method and machine learning technique are summarized
 218 in Table 2. Complete results (all the non-dominated solutions separated by datasets) are
 219 provided in the supplementary document.

220 Rather than discussing all 12 datasets separately, we categorize our existing datasets into
 221 three clusters as small-, medium-, and large-sized datasets, in order to make comparisons
 222 easier. For this purpose, we sorted all the datasets by their number of features in ascending
 223 order. Categorizing PM, NU, and BC as small-sized datasets was straightforward due to
 224 their small number of features (up to 9). For deciding on where to split the medium- and
 225 large-sized datasets, the increment in the number of features from 42 (C4) to 54 (CT) stands
 226 out at first glance. However, the huge difference in the number of instances between WF
 227 (5000) to C4 (67,557) is also remarkable, and C4 resembles CT (581,012) more on this
 228 aspect. As a result, even though the number of features between WF and C4 differentiates
 229 by only 2, we decided to categorize WF as a medium-sized dataset, and C4 as a large-sized
 230 dataset, due to their number of instances. Hence, we obtain the clusters as specified below:
 231

- 232 • small-sized datasets: PM, NU, and BC
- 233 • medium-sized datasets: MR, WBC, IO, and WF
- 234 • large-sized datasets: C4, CT, SB, FI, and MU

235 We would like to note that our categorization is not a representative of a universal cat-
 236 egorization considering enormous datasets used in other domains such as text mining or
 237 bioinformatics. We categorize some of our datasets as large since they are larger than the
 238 others in this study only.

239 Obtained results indicate that, on the small-sized datasets, all methods can decrease the
 240 number of features with similar accuracy values. The execution time of IGR is less than
 241 other methods in most cases. For medium-sized datasets, it is clear that Gr performs well in

Table 3: Mean and standard deviation of the dataset accuracies with LR classification.

Dataset	Rnd	Gr	IGR	DR	SR
CT	0.741 ± 0.064	0.768 ± 0.012	0.762 ± 0.011	0.771 ± 0.006	0.716 ± 0.070
MR	0.807 ± 0.169	0.904 ± 0.070	0.859 ± 0.147	0.900 ± 0.073	0.831 ± 0.128
SB	0.896 ± 0.022	0.897 ± 0.022	0.854 ± 0.052	0.900 ± 0.017	0.826 ± 0.076
NU	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.807 ± 0.272
C4	0.794 ± 0.012	0.787 ± 0.024	0.782 ± 0.021	0.788 ± 0.025	0.739 ± 0.005
WF	0.904 ± 0.022	0.893 ± 0.040	0.905 ± 0.024	0.912 ± 0.014	0.837 ± 0.126
FI	0.966 ± 0.000	0.966 ± 0.000	0.966 ± 0.000	0.953 ± 0.018	0.966 ± 0.000
PM	0.765 ± 0.011	0.767 ± 0.011	0.765 ± 0.011	0.767 ± 0.011	0.737 ± 0.052
BC	0.954 ± 0.018	0.955 ± 0.016	0.955 ± 0.016	0.950 ± 0.021	0.939 ± 0.043
IO	0.844 ± 0.086	0.896 ± 0.008	0.840 ± 0.015	0.838 ± 0.092	0.835 ± 0.089
WBC	0.920 ± 0.115	0.964 ± 0.024	0.967 ± 0.014	0.956 ± 0.026	0.921 ± 0.105
MU	0.930 ± 0.004	0.891 ± 0.022	0.896 ± 0.027	0.935 ± 0.004	0.856 ± 0.009

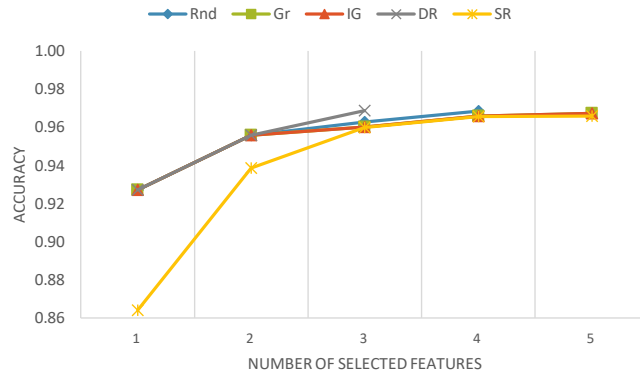
242 terms of both minimizing the number of features and maximizing accuracy. Rnd and IGR
 243 compete with each other, and both dominate DR. Similar to small-sized datasets, execution
 244 times of IGR are much less than the execution times of the others. The result for the
 245 large-sized datasets is interesting. Both Rnd and DR are outperformed by Gr and IGR in
 246 terms of minimizing the number of features. Among Gr and IGR, Gr achieves more diverse
 247 non-dominated solutions than IGR does. Again, IGR is the fastest method among all of
 248 them. Also, results show that SR can reduce the number of features well, yet, it struggles
 249 to find high accuracy values in most cases.

250 This analysis shows us that, Rnd and DR are competitive initial population generation
 251 methods in small-sized datasets, however, they tend to get stuck in local optima as datasets
 252 grow larger. This outcome could be triggered by the number of selected features in the initial
 253 population. Gr and IGR algorithms begin selecting the number of features from 1, up to
 254 population size in the initial population. This is actually a good starting point considering
 255 that one of the objectives is reducing the number of features. In medium- and large-sized
 256 datasets, Rnd and DR can only achieve high accuracy values with a higher number of features
 257 than other methods (Gr, IGR, and SR).

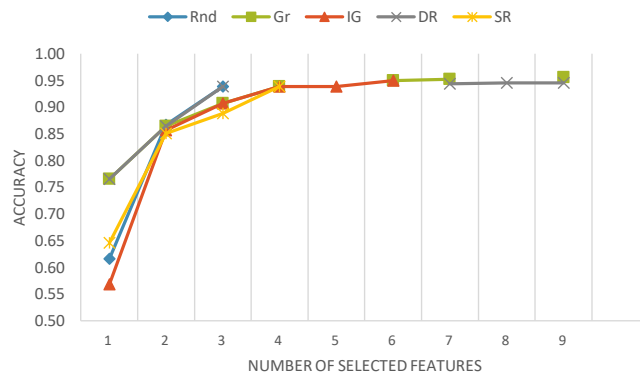
258 To carry out a more complete comparison, we give mean and standard deviation of
 259 the accuracy values of datasets for all initial population generation methods when LR is
 260 applied, in Table 3. Moreover, Figure 4 presents the non-dominated solution sets of each
 261 initial population generation method on three selected datasets when LR is applied.

262 To provide more robust analysis, we enhance the comparison results with statistical tests.
 263 We use Friedman’s Test to test differences of accuracy, the number of features and execution
 264 time between initial population generation methods, which indicates a significant difference
 265 in every comparison. A post hoc Wilcoxon signed-rank test with Bonferroni Correction is
 266 employed for pairwise comparison, with setting the significance level at $p < 0.01$, and the
 267 results are presented in Table 4.

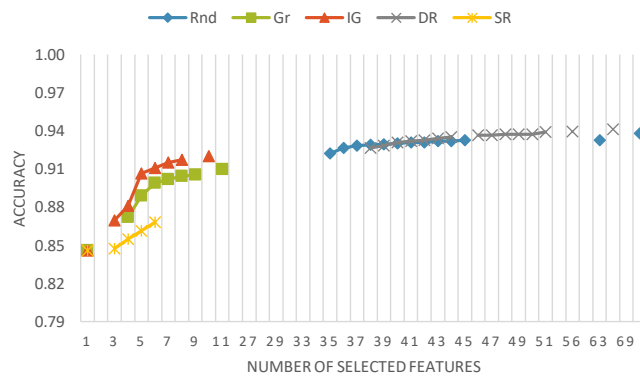
268 Statistical tests indicate no significant difference between initial population generation
 269 methods in terms of maximizing accuracy. The only exception is the SR method, which per-



(a) Non-dominated solutions on BC dataset evaluated by LR.



(b) Non-dominated solutions on MR dataset evaluated by LR.



(c) Non-dominated solutions on MU dataset evaluated by LR.

Figure 4: Non-dominated solutions evaluated by LR on three datasets (one to represent each small, medium and large-sized datasets) using five different initial population generation methods.

Table 4: Results of post hoc pairwise Wilcoxon signed-rank test with Bonferroni Correction (significance level set to $p < 0.01$). Friedman’s test results for each group are given in parantheses under each machine learning technique. Significant differences are presented in **bold**.

(a) Accuracy

	LR ($\chi^2(4) = 19.705, p = 0.001$)				SVM ($\chi^2(4) = 23.280, p < 0.001$)				ELM ($\chi^2(4) = 17.745, p = 0.001$)			
	Rnd	Gr	IGR	DR	Rnd	Gr	IGR	DR	Rnd	Gr	IGR	DR
Gr	Z=-1.173 p=0.241	-	-	-	Z=-0.700 p=0.484	-	-	-	Z=-0.714 p=0.475	-	-	-
IGR	Z=-0.296 p=0.767	Z=-1.304 p=0.192	-	-	Z=-0.841 p=0.400	Z=-0.070 p=0.944	-	-	Z=-0.204 p=0.838	Z=-0.771 p=0.441	-	-
DR	Z=-1.023 p=0.306	Z=-0.459 p=0.646	Z=-1.201 p=0.230	-	Z=-0.210 p=0.833	Z=-0.663 p=0.507	Z=-1.020 p=0.308	-	Z=-0.306 p=0.760	Z=-0.889 p=0.374	Z=-0.631 p=0.528	-
SR	Z=-2.490 p=0.013	Z=-2.934 p=0.003	Z=-2.941 p=0.003	Z=-2.824 p=0.005	Z=-2.934 p=0.003	Z=-2.847 p=0.004	Z=-2.936 p=0.003	Z=-2.934 p=0.003	Z=-2.934 p=0.003	Z=-2.845 p=0.004	Z=-2.667 p=0.008	Z=-2.401 p=0.016

(b) Number of features

	LR ($\chi^2(4) = 12.905, p = 0.012$)				SVM ($\chi^2(4) = 11.752, p = 0.019$)				ELM ($\chi^2(4) = 20.451, p < 0.001$)			
	Rnd	Gr	IGR	DR	Rnd	Gr	IGR	DR	Rnd	Gr	IGR	DR
Gr	Z=-1.424 p=0.154	-	-	-	Z=-2.666 p=0.008	-	-	-	Z=-2.805 p=0.005	-	-	-
IGR	Z=-1.836 p=0.066	Z=-1.943 p=0.052	-	-	Z=-2.310 p=0.021	Z=-0.140 p=0.889	-	-	Z=-2.670 p=0.008	Z=-0.511 p=0.610	-	-
DR	Z=-1.782 p=0.075	Z=-2.497 p=0.013	Z=-2.599 p=0.009	-	Z=-1.274 p=0.203	Z=-2.310 p=0.021	Z=-2.293 p=0.022	-	Z=-1.367 p=0.172	Z=-2.666 p=0.008	Z=-2.670 p=0.008	-
SR	Z=-1.929 p=0.054	Z=-2.253 p=0.024	Z=-1.252 p=0.211	Z=-2.669 p=0.008	Z=-2.825 p=0.005	Z=-1.067 p=0.286	Z=-1.156 p=0.248	Z=-2.224 p=0.026	Z=-2.395 p=0.017	Z=-0.045 p=0.964	Z=-0.939 p=0.348	Z=-1.993 p=0.046

(c) Execution time

	LR ($\chi^2(4) = 28.533, p < 0.001$)				SVM ($\chi^2(4) = 20.867, p < 0.001$)				ELM ($\chi^2(4) = 42.200, p < 0.001$)			
	Rnd	Gr	IGR	DR	Rnd	Gr	IGR	DR	Rnd	Gr	IGR	DR
Gr	Z=-0.628 p=0.530	-	-	-	Z=-2.275 p=0.023	-	-	-	Z=-2.275 p=0.023	-	-	-
IGR	Z=-2.903 p=0.004	Z=-2.903 p=0.004	-	-	Z=-2.510 p=0.012	Z=-1.726 p=0.084	-	-	Z=-3.059 p=0.002	Z=-3.059 p=0.002	-	-
DR	Z=-2.903 p=0.004	Z=-1.177 p=0.239	Z=-2.903 p=0.004	-	Z=-1.255 p=0.209	Z=-1.177 p=0.239	Z=-2.510 p=0.012	-	Z=-3.059 p=0.002	Z=-2.903 p=0.004	Z=-3.059 p=0.002	-
SR	Z=-2.981 p=0.003	Z=-2.981 p=0.003	Z=-0.784 p=0.433	Z=-2.981 p=0.003	Z=-2.275 p=0.023	Z=-2.118 p=0.034	Z=-0.784 p=0.433	Z=-2.510 p=0.012	Z=-3.059 p=0.002	Z=-3.059 p=0.002	Z=-0.157 p=0.875	Z=-3.059 p=0.002

270 forms worse than all others, in almost all cases. In the comparison of the number of features:
271 IGR and SR methods perform better than DR for LR; Gr and SR perform better than Rnd
272 for SVM; Gr and IGR perform better than Rnd and DR for ELM. Moreover, the difference
273 is only marginally insignificant in 6 cases ($p < 0.025$). We conclude that Gr, IGR and SR
274 perform better than Rnd and DR in terms of reducing the number of features. Next, we
275 compare execution times of the initial population generation methods. Results indicate that
276 IGR and SR methods execute faster than Rnd, Gr and DR methods, in general. SVM, at

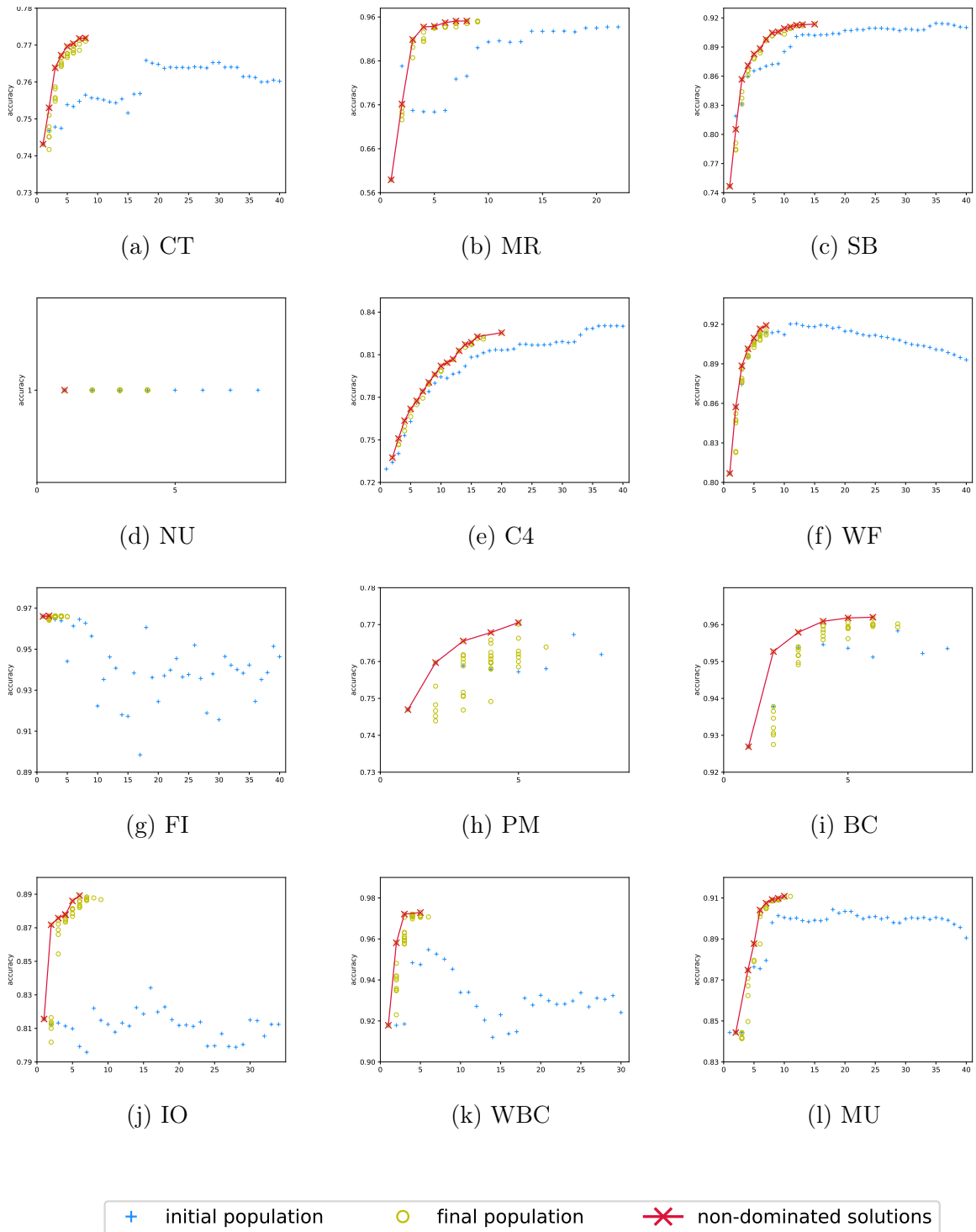


Figure 5: Initial population and final population distributions of the MTLBO-MD + IGR (population size=40, number of generations=60).

Table 5: Training and test set sizes of datasets for comparison with previous studies.

	CT	MR	SB	NU	C4	WF	FI	PM	BC	IO	WBC	MU
Training set	600	1300	600	400	1200	400	1000	268	199	101	169	400
Test set	200	200	200	200	200	200	200	200	100	50	80	200

277 first glance, seems like an exceptional case with no significant differences between methods.
 278 However, even though pairwise differences in SVM is not significant, Friedman’s test indi-
 279 cates a significant difference between initial population generation methods. Furthermore,
 280 the difference between IGR and Rnd, IGR and DR, and, SR and DR is only marginally
 281 insignificant, with a p-value of 0.012.

282 Analyzing the results, it is clear that SR cannot perform well in terms of accuracy.
 283 Among the remaining 4 initial population generation methods, Gr and IGR perform better
 284 than Rnd and DR, in terms of minimizing the number of features. As a final comparison
 285 to decide between Gr and IGR, we see that the execution time of IGR is better than the
 286 execution time of Gr. From these results, we conclude that the IGR method achieves good
 287 enough solutions in a timely manner, standing as a promising method for initial population
 288 generation. We also present the performances of initial and final populations for each dataset
 289 in Figure 5, when the IGR method is employed.

290 Finally, we compare the results of the IGR initial population generation method with
 291 results obtained from our previous studies, where MTLBO-MD (Kiziloz et al., 2018) and
 292 NSGA-II (Deniz et al., 2017) were used as metaheuristics with LR as the classifier and
 293 the initial population was generated randomly. However, in those studies, population size
 294 and number of generations were twice as much (population size was 40 and the number of
 295 generations was 60). Also, they used another method for splitting data into training and
 296 test sets, similar to other studies in the literature (Pacheco et al., 2009; Unler & Murat,
 297 2010). Briefly, for each dataset, they calculated instance proportion per class, and randomly
 298 selected instances from classes with respect to their proportion to fill up the training set sizes
 299 given in Table 5. Then, they randomly selected instances, apart from those existing in the
 300 training set, for the test set. Hence, we mimicked this methodology for all datasets prior to
 301 the study and generated 10 training set and 10 test set per each training set (a total of 100
 302 test sets). Similar to those studies, we increased the population size to 40 and the number
 303 of generations to 60, reran our tests of IGR method with LR selected as the classifier.
 304 Comparison results for all 12 datasets are given in Table 6. IGR produces close enough
 305 solutions to randomly initialized MTLBO-MD and NSGA-II algorithms. Numerically, out
 306 of 73 available comparisons (a row in the table, where MTLBO-MD + IGR method may
 307 be compared with at least one other method in terms of accuracy value, is considered as an
 308 available comparison), IGR can find the maximum accuracy value in 36 occasions. It finds
 309 a solution with up to 0.005 decrease in maximum accuracy in 60 occasions. Moreover, it
 310 requires less amount of time to execute in medium and large-sized datasets.

311 Possible limitations to the study could be specified as follows. The most important
 312 drawback is the specific domain we evaluate our results on, namely FSS. FSS is an important
 313 step for building an expert system, yet it is not considered as one by itself. Hence, we do

314 not directly evaluate its effect on a real expert system. Secondly, the selected crossover and
 315 mutation operators have their own working schemes, which may affect the outcome of the
 316 study. Likewise, other types of search algorithms could lead to different results.

317 4. Conclusion

318 Machine learning applications may suffer from high dimensional data. As the number of
 319 redundant and/or irrelevant features in a dataset increase, learning time and accuracy are
 320 affected badly. Feature Subset Selection (FSS) is an important tool for dimension reduction,
 321 and evolutionary based heuristic algorithms are utilized for this purpose. These algorithms
 322 require an initial population to begin their search, and initial population selection may affect
 323 the final solution.

324 In this study, we aim to improve the performance of an expert system, by improving the
 325 performance of an intermediate step, FSS. To improve the FSS performance, we propose
 326 selecting its initial population wisely. For this reason, we compare five initial population se-
 327 lection methods, namely Random (Rnd), Distinct Random (DR), Information Gain Ranking
 328 (IGR), Greedy (Gr), and Small Random (SR). We use the MTLBO-MD heuristic algorithm
 329 and three machine learning techniques, namely Logistic Regression, Support Vector Ma-
 330 chines, and Extreme Learning Machine, for evaluating the performance of the solutions.
 331 In our experiments, we utilize a total of 12 publicly available datasets, most of which are

Table 6: Multiobjective comparison of the IGR method with previous studies.

(a) C4				(b) SB				(c) MU			
F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd	F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd	F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd
1	-	0.729	-	1	0.747	0.782	-	2	0.844	-	-
2	0.737	0.746	-	2	0.805	0.835	0.835	4	0.875	-	-
3	0.751	0.755	-	3	0.857	0.857	0.857	5	0.888	-	-
4	0.764	0.764	0.765	4	0.871	0.871	0.871	6	0.904	-	-
5	0.772	0.772	0.772	5	0.883	0.879	0.883	7	0.907	-	-
6	0.778	0.777	0.778	6	0.888	0.891	0.890	8	0.909	-	-
7	0.784	0.784	0.785	7	0.898	0.896	0.902	9	0.910	-	-
8	0.791	0.791	0.791	8	0.905	0.905	0.906	10	0.911	-	-
9	0.796	0.796	0.796	9	0.906	0.910	0.910	21	-	0.907	-
10	0.802	0.797	0.800	10	0.909	0.911	0.914	22	-	0.910	-
11	0.804	0.799	0.802	11	0.911	0.913	0.915	23	-	0.912	-
12	0.807	0.799	0.807	12	0.913	-	0.916	24	-	0.914	-
13	0.813	-	0.810	13	0.913	0.915	0.917	25	-	0.916	-
14	0.817	-	0.812	14	-	-	0.919	26	-	0.917	-
15	0.819	-	0.815	15	0.914	-	0.919	27	-	0.918	-
16	0.823	0.805	0.819	16	-	-	0.920	28	-	0.919	-
17	-	-	0.821	17	-	-	0.920				
19	-	-	0.822	18	-	-	0.920				
20	0.826	-	0.822								
22	-	-	0.825	Time	372.1	426.1	1171.8	Time	76.9	931.2	-
23	-	-	0.826								
24	-	-	0.826								
Time	502.6	431.6	1197.6								

(d) CT

F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd
1	0.743	0.743	0.743
2	0.753	0.753	0.753
3	0.764	0.764	0.764
4	0.767	0.767	0.767
5	0.770	0.770	0.770
6	0.770	0.772	0.772
7	0.772	0.773	0.772
8	0.772	0.773	0.773
9	-	0.774	0.774
10	-	-	0.774
13	-	-	0.774
14	-	-	0.775
Time	170.4	548.7	964.4

(e) WF

F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd
1	0.807	0.789	0.868
2	0.857	0.868	0.893
3	0.889	0.893	0.902
4	0.901	0.902	0.915
5	0.910	0.915	0.917
6	0.917	0.917	0.919
7	0.919	0.919	0.921
8	-	0.921	0.922
9	-	0.922	0.923
10	-	0.923	0.923
Time	47.9	88.8	176.6

(f) MR

F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd
1	0.589	0.763	0.750
2	0.762	0.905	0.867
3	0.909	0.937	0.937
4	0.937	0.940	0.937
5	0.939	0.949	0.945
6	0.947	0.950	0.946
7	0.951	-	-
8	0.951	-	0.946
9	-	-	0.949
Time	211.3	158.8	303.9

(g) IO

F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd
1	0.816	0.816	-
2	0.872	0.872	0.872
3	0.876	0.876	0.876
4	0.878	0.882	0.878
5	0.886	0.886	0.886
6	0.889	0.886	0.889
7	-	0.887	-
8	-	0.890	-
Time	85.1	131.6	239.6

(h) BC

F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd
1	0.927	0.927	0.927
2	0.953	0.953	0.953
3	0.958	0.963	0.963
4	0.961	0.963	0.963
5	0.962	0.963	0.963
6	0.962	-	-
Time	11.1	7.2	12.4

(i) PM

F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd
1	0.747	0.747	0.747
2	0.760	0.760	0.760
3	0.766	0.766	0.766
4	0.768	0.768	0.768
5	0.771	0.771	0.770
Time	4.8	4.9	5.9

(j) WBC

F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd
1	0.918	0.920	0.920
2	0.958	0.961	0.961
3	0.972	0.971	0.972
4	-	0.975	0.975
5	0.973	-	-
Time	35.5	54.1	89.2

(k) FI

F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd
1	0.966	0.966	0.966
2	0.966	-	-
3	-	0.967	0.966
Time	115.8	702.8	2721.7

(l) NU

F. size	MTLBO MD + IGR	MTLBO MD + Rnd	NSGA II + Rnd
1	1.000	1.000	1.000
Time	22.0	12.5	17.5

332 obtained from the UCI Machine Learning Repository.

333 Experimental results indicate that all initial population generation methods can find
334 good enough solutions in a timely manner when dataset size is small. As dataset size grows
335 larger, random based methods get stuck in local optima and cannot compete with Gr and
336 IGR. Among these two algorithms, Gr achieves slightly better results than IGR. On the other
337 hand, the execution time of Gr is enormous as compared to the IGR method. Additionally,
338 the IGR method is compared with two other algorithms in literature having random initial-
339 izations. The obtained results show that the IGR method can achieve comparable solutions

340 with smaller execution time. As a result, we suggest using IGR as the initial population
341 generation method for FSS, for small, medium and large-sized datasets.

342 The main strength of the proposed method is its simplicity. Information Gain is a well-
343 known algorithm, and its implementation in our method is straightforward. On the other
344 hand, this method is not readily available in many existing frameworks, hence an additional
345 step is required for its implementation.

346 The main contribution of this study can be summarized as follows. Even though the
347 importance of initial population generation is theoretically well-known, initializing the pop-
348 ulation randomly is still the common approach. We analyze five different initial popula-
349 tion generation methods with empirical results to emphasize the subject and find the most
350 promising method. Our experiment results clearly show that generating the initial popula-
351 tion wisely improves the overall performance, for all dataset sizes.

352 Possible future work for this study can be implementing different types of initial popula-
353 tion generation methods, such as a hybrid method that combines information gain, greedy
354 approach and/or randomness. Also, other existing methodological settings can be tested.
355 For example, different searching algorithms such as NSGA-II or PSO may be used instead
356 of MTLBO-MD. Similarly, different types of crossover and mutator operators exist in the
357 literature. We used half uniform crossover and bit-flip mutation, respectively, and it could
358 be interesting to see whether the obtained results generalize over all operators. Moreover,
359 in this study, we verified the importance of generating the initial population wisely, for the
360 FSS domain. Wise initial population generation approaches for other problem domains, such
361 as bin packing, graph coloring, travelling salesman, etc., can also be investigated. Finally,
362 initial population generation methods can be implemented specifically to different dataset
363 types such as image, text, etc.

364 References

- 365 BBC (March 2014). Big data: Are you ready for blast-off? [http://www.bbc.com/news/](http://www.bbc.com/news/business-26383058)
366 [business-26383058](http://www.bbc.com/news/business-26383058). [Online; accessed 20-May-2017].
- 367 Deng, Y., Liu, Y., & Zhou, D. (2015). An improved genetic algorithm with initial population strategy for
368 symmetric tsp. *Mathematical Problems in Engineering, 2015*.
- 369 Deniz, A., Kiziloz, H. E., Dokeroglu, T., & Cosar, A. (2017). Robust multiobjective evolutionary feature
370 subset selection algorithm for binary classification using machine learning techniques. *Neurocomputing,*
371 *241*, 128–146.
- 372 Diaz-Gomez, P. A., & Hougen, D. F. (2007). Initial population for genetic algorithms: A metric approach.
373 In *GEM* (pp. 43–49).
- 374 Dua, D., & Graff, C. (2017). UCI machine learning repository.
- 375 Escobar, A. H., Gallego, R. A., & Romero, R. A. (2011). Using traditional heuristic algorithms on an
376 initial genetic algorithm population applied to the transmission expansion planning problem. *Ingeniería*
377 *e Investigación, 31*, 127.
- 378 Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine*
379 *learning research, 3*, 1157–1182.
- 380 Hill, R. R. (1999). A monte-carlo study of genetic algorithm initial population generation methods. In
381 *Proceedings of the 31st conference on Winter simulation: Simulation—a bridge to the future-Volume 1*
382 (pp. 543–547). ACM.

- 383 Horn, J., Nafpliotis, N., & Goldberg, D. E. (1994). A niched pareto genetic algorithm for multiobjective
384 optimization. In *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence.,*
385 *Proceedings of the First IEEE Conference on* (pp. 82–87). Ieee.
- 386 Huang, G.-B., Wang, D. H., & Lan, Y. (2011). Extreme learning machines: a survey. *International Journal*
387 *of Machine Learning and Cybernetics*, 2, 107–122.
- 388 Jorapur, V. S., Puranik, V. S., Deshpande, A. S., & Sharma, M. (2016). A promising initial population based
389 genetic algorithm for job shop scheduling problem. *Journal of Software Engineering and Applications*, 9,
390 208.
- 391 Karci, A. (2004). Novelty in the generation of initial population for genetic algorithms. In *International*
392 *Conference on Knowledge-Based and Intelligent Information and Engineering Systems* (pp. 268–275).
393 Springer.
- 394 Kiziloz, H. E., Deniz, A., Dokeroglu, T., & Cosar, A. (2018). Novel multiobjective tlbo algorithms for the
395 feature subset selection problem. *Neurocomputing*, .
- 396 Maaranen, H., Miettinen, K., & Penttinen, A. (2007). On initial populations of a genetic algorithm for
397 continuous optimization problems. *Journal of Global Optimization*, 37, 405.
- 398 Michalewicz, Z. (1996). Evolution strategies and other methods. In *Genetic algorithms+ data structures=*
399 *evolution programs* (pp. 159–177). Springer.
- 400 Pacheco, J., Casado, S., & Núñez, L. (2009). A variable selection method based on tabu search for logistic
401 regression models. *European Journal of Operational Research*, 199, 506–511.
- 402 Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, 1, 81–106.
- 403 Toğan, V., & Daloglu, A. T. (2008). An improved genetic algorithm with initial population strategy and
404 self-adaptive member grouping. *Computers & Structures*, 86, 1204–1218.
- 405 Unler, A., & Murat, A. (2010). A discrete particle swarm optimization method for feature selection in binary
406 classification problems. *European Journal of Operational Research*, 206, 528–539.
- 407 Zhou, X., Miao, F., & Ma, H. (2018). Genetic algorithm with an improved initial population technique for
408 automatic clustering of low-dimensional data. *Information*, 9, 101.