Authors' Statement:

We have two updates on our published paper entitled "**A hybrid feature selection algorithm based on large neighborhood search**" as below:

(1) **A minor revision on the paper:** According to the section 3.2 of the paper, the destroy function takes those features into account that have been set to one, while it actually considers all features which have been set to either zero or one. So, the following revises applied on the paper:

- Page 6, line 4, will be replaced with the below sentence after revision:

" xd = des(x) : Select K percentage of the lowest ranked features (either zero or one) to remove from x "

- Page 6, the sentence at line 38, will be replaced with the below sentence after revision:

"In order to destruct the suitable part of the solution, we employed CFS to rank features and remove lowest rank K percentage of features which were set to either zero or one."

- Page 7, the sentence at line 6, will be replaced with the below sentence after revision:

"As mentioned before, function des() selects K percentage of the lowest ranked features (which are either zero or one) from x , using the CFS ranking selection and move it to xd"

(2) **A new added appendix**: The appendix provides subsets of selected features by the proposed algorithm for each data set for five independent runs.

(3) **The implementation of the paper**: The implementation has been uploaded on the following GitHub repository: *https://github.com/gelarehai/FeatureSelection*

A Hybrid Feature Selection Algorithm Based on Large Neighborhood Search

Gelareh Taghizadeh and Nysret Musliu

Database and Artificial Intelligence Group,
Vienna University of Technology,
Vienna, Austria,
{gtaghiza,musliu}@dbai.tuwien.ac.at

Abstract. Feature selection aims at choosing a small number of relevant features in a data set to achieve similar or even better classification accuracy than using all features. This paper presents the first study on Large Neighborhood Search (LNS) algorithm for the feature selection problem. We propose a novel hybrid Wrapper and Filter feature selection method using LNS algorithm (WFLNS). In LNS, an initial solution is gradually improved by alternately destroying and repairing the solution. We introduce the idea of using filter ranking method in the process of destroying and repairing to accelerate the search in identifying the core feature subsets. Particularly, WFLNS either adds or removes features from a candidate solution based on the correlation based feature ranking method. The proposed algorithm has been tested on twelve benchmark data sets and the results have been compared with ten most recent wrapper methods where WFLNS outperforms other methods in several the data sets.

Keywords: Feature Selection, Large Neighborhood Search, Classification

1 Introduction

One of the aims of the feature selection is to remove irrelevant and redundant features from a set of original features to improve the classification performance. Generally, recent interest in feature selection has been increased due to challenges which are caused by sheer volume of data that is expected to have a rapid growth over next years. Such data volume would not only increase the demand of computational resources but also effect on the quality of several data mining tasks such as classification. Moreover, learning from large data sets would be a more complex task when it includes irrelevant, redundant and noisy features. Feature selection techniques address such challenges by decreasing the dimensionality, reducing the amount of data needed for the learning process, shortening the running time, and improving the performance of the learnt classifiers [1].

Throughout the literature, feature selection approaches are mainly categorized into three main groups: filter, wrapper and hybrid approaches. Filter approaches select and evaluate subsets of features based on the general characteristics of the training data without involving a learning model while wrapper approaches find subset of features by considering a learning model. Wrapper methods have better quality subsets than filter methods, however they impose a high computational cost as a result of constructing a learning model when evaluating every single subset while filter methods are more computationally efficient. Hybrid approaches aim at benefiting from advantages of both filter and wrapper methods, where filter method can search through the feature space efficiently while the wrapper method provides good accuracy by providing higher quality subset of features for a classifier.

Wrapper methods guide the search through the search space of feature subset by a learning algorithm where a search algorithm is wrapped around the classification model. To this end a common open question that all wrapper techniques try to deal with is to develop an efficient search algorithm for finding an optimal feature subset. An exhaustive search algorithm may be employed, as a possible option to find the optimal solution, however it is often impractical for most data sets. Alternatively, metaheuristic methods have been developed with varying degrees of success (see the next section) to find near-optimal solutions in a reasonable amount of time.

This paper proposes a novel hybrid Wrapper and Filter algorithm designed based on LNS (WFLNS) to solve the feature selection problem. In this study we aim at developing a LNS with improved capabilities of destroy, repair and acceptance methods that has leading-edge performance over the most recent metaheuristics algorithms for feature selection. Our idea is based on hybridizing a filter ranking method as a heuristic in the process of destroying and repairing of the LNS to transform a current solution into a different solution by constantly adding or removing features based on the correlation based feature ranking method [2]. Moreover, we proposes a new acceptance method in WFLNS by incorporating the Simulated Annealing acceptance probability. It is worth mentioning that to the best of our knowledge no systematic studies have been carried out to investigate the capability of LNS algorithms for the feature selection problem so far. The performance of the proposed algorithm is compared with ten state of the art metaheuristic algorithms which all were employed for solving the feature selection problem in [3]. Moreover, we used the same high dimensional and real-valued benchmark data sets as [3], which were originally collected from UCI repository [4]. We were able to conclude that our WFLNS outperforms other algorithms in terms of classification accuracy, particularly for large size data set. The rest of the paper is organized: Section 2 reviews the prior studies in metaheuristic algorithms for feature selection problem. Section 3 proposes Large Neighborhood Search for the feature selection problem and experimental results and discussions are presented in Section 4. Finally, Section 5 concludes the paper.

2 Literature Review

Feature selection problem is the problem of choosing the best subset of features out of the whole feature set. Searching for the best subset in the feature space requires checking all possible combinations of features in the search space. To this end, feature selection problem could be considered as a NP-hard problem [5]. Metaheuristic algorithms are introduces into feature selection as an appropriate methods as they have been demonstrated to be superior methodologies in many NP hard problems. As the core concept of WFLNS is close to the concepts of both wrapper and hybrid approaches, here we concentrated on the literature that has focused on wrapper and hybrid feature selection by employing metaheuristic algorithms. Different feature selection algorithms have been developed based on different sort of optimization techniques such as metaheuristic algorithms. We explored and categorized the existing metaheuristic-based feature selection algorithms into three groups: trajectory- based feature selection, population-based feature selection and hybrid feature selection. A trajectory-based algorithm typically uses one solution at a time, which will gradually improve the current solution by constantly changing the current solution as the iterations continue. The trajectory-based algorithms developed for feature selection problem include Tabu Search (TS) [6] [7] , Simulated Annealing (SA) [8] and Harmony Search (HS) [9] . In [10], a hybrid approach based on TS and probabilistic neural networks is proposed for the feature selection problem. In contrast with other TS based approached for the feature selection problem, this approach employed long-term memory to avoid the necessity of tuning the memory length and decrease the risk of trapping into local optimal solutions. A TS was employed in [11] to select effective emotion states from the physiological signals based on K-nearest neighbor classifier. In [12], a SA-SVM approach is developed for tuning the parameter values of SVM and finding the best subset of features in a way that maximize the classification accuracy of SVM. Another SA-based algorithm is proposed in [13], where feature selection applied on marketing data to build large scale regression model. In the proposed approach, SA was compared with stepwise regression [14] (as a typical example of an iterative improvement algorithm) and the results have shown the superiority of SA in providing a better predictive model by escaping the local optimum that stepwise regression is fall into. In [15], hybrid combination of SA and GA proposed to combine the capability of SA and GA to avoid being trapped in a local minimum and benefit from the high rate of convergence of the crossover operator of genetic algorithms in the search process. In [16] HS is employed for feature selection in email classification task, where HS was incorporated with the fuzzy support vector machine and Naive Bayesian classifiers.In [17], the authors propose a hybrid feature selection algorithm based on the filter and HS algorithm for gene selection in micro-array data sets

Population-based approaches rely on a set of candidate solutions rather than on one current solution. There exist two main population based algorithms, which are developed for feature selection problem, evolutionary algorithms and swarm intelligence algorithms. Former involves Genetic Algorithms (GA) [18], Memetic Algorithms (MA) [19] and Artificial Immune Algorithms [20] while later referring to Particle Swarm Optimization (PSO) [21], Ant Colony Optimization (ACO) [22], and Artificial Bee Colony (ABC) [23] optimization. In [24] a GA-base feature selection was proposed in which a feature subset is represented by a binary string of length of the total number of features, called a chromosome. A population of such chromosomes is randomly initialized and maintained, and those with higher classification accuracy are propagated into the later generations. In [25], a new hybrid algorithm of GA and PSO was developed for classification of hyper spectral data set based on SVM classifier. [26] developed a feature selection algorithm based on MA for multi-label classification problem which prevent premature convergence by employing a local search to refine the feature subsets found through a GA search. A hybrid filter-wrapper method based on memetic framework was proposed in [27], which employes the filter-ranking method at each iteration of the MA to rank the features and reduce the neighborhood size, then local search operators are applied. Clonal Search Algorithm (CSA) is inspired from Artificial Immune Algorithm, which was developed for the feature selection problem by Shojaei and Moradi in [28] where the proposed approach enables both feature selection and parameter tuning of the SVM classifier. Among swarm intelligence metaheuristic algorithms, the ACO-based feature selection approach proposed in [29] that each ant has a binary vector where 0 and 1 represents deselected and selected features respectively. [30] presents a variation of ACO for the feature selection problem, which is called enriched ACO. It aims at considering the previously traversed edges in the earlier executions to adjust the pheromone values appropriately and prevent premature convergence of the ACO. [31] proposed a rough set-based binary PSO algorithm to perform feature selection. In the algorithm, each particle represents a potential solution, and these are evaluated using the rough set dependency degree. An ABC algorithm was proposed in [32] to solve multi-objective feature selection problem where the number of features should be minimized while the classification accuracy should be maximized. The results where evaluated based on mutual information, fuzzy mutual information and the proposed fuzzy mutual information.

In order to improve the searching ability of proposed feature selection algorithms, hybrid algorithms were proposed. The most effective forms of the combination is to use both filter and wrapper approach at the same time. Generally, hybrid approaches involve two main steps. At the first step a filter method applies to reduce the number of features and consequently the searching space. The second step is a wrapper method that explores the subsets, which were built on the first step. In [33] a combination of information gain as a filter method and GA as a wrapper method was proposed. The K-nearest neighbor (KNN) classifier with leave-one-out cross-validation (LOOCV) employed as an evaluator of the proposed algorithm. Another hybrid approach is developed for developing short term forecasting in [34], which first uses partial mutual information based filter method to remove most irrelevant features, and subsequently applies a wrapper method through firefly algorithm to further reduce the redundant features without degrading the forecasting accuracy. Another hybrid approach was developed in [35] for feature selection in DNA micro-arrays data set. The proposed algorithm employed both univariate and multivariate filter methods along with the GA as a wrapper method where the reported results show that a multivariate filter outperforms a univariate filter in filterwrapper hybrid. [36] presents the application of rough set method on the outcome of Principal Components Analysis (PCA) for the feature selection problem on neural network classifiers for a face image data set. Greedy Randomized Adaptive Search Procedure (GRASP) is a multi-start two-phase algorithm, construction and local search phase. A feasible solution is built in the construction phase, and then its neighborhood is explored by the local search. [37] proposes a GRASP-based algorithm for the feature selection problem, where the proposed algorithm uses some random part of the features subset at each iteration, and then selects features based on cooperation between all the previously found non-dominated solutions. [38] is investigated binary classification high dimension data sets while employing GRASP algorithm for feature selection to reduce the computation time. In [39] a feature selection approach is proposed based on a linear programming model with integer variables on biological application. To deal with such approach the authors presents a metaheuristic algorithm based on GRASP procedure which is extended with the adoption of short memory and a local search strategy. The reported results show that the method performs well on a very large number of binary or categorical features. In [3] authors carried out a comprehensive review on the most recent metaheuristic algorithms for the feature selection problem. The performance of developed algorithms were examined and compared with each other based on twelve benchmark data set from UCI repository. We considered this paper as a base line for evaluating our proposed method, and compare our experiments with their reviewed algorithms with the same setting and data sets.

3 WFLNS:A Wrapper Filter feature selection based on LNS

LNS algorithm [40] is a metaheuristic search algorithm, which aims at finding a near-optimal solution by iteratively transforming a current solution to an alternative solution in its neighborhood. The notion of the neighborhood in LNS algorithm refers to a set of similar candidate solutions, which is achieved by applying destroy and repair methods over the current solution [41]. Using large neighborhoods makes it possible to find better candidate solutions at each iteration of the algorithm and hence explore more promising part of the search space. LNS-based algorithms have been applied on many optimization problems, including the traveling salesman problem space [42], timetabling problems [43] [44] and capacitated vehicle routing problem [44] because of their capabilities to explore a wide samples of the search space and escape from local optima by means of destroy and repair techniques along with an appropriate acceptance criterion. To the best of our knowledge no systematic studies have been carried out to investigate the capability of LNS-based algorithms for the feature selection problem. Here we present WFLNS, a hybrid Wrapper and Filter feature selection algorithm based on the LNS framework. The novelty of the WFLNS includes embedding new problem-specific destroy, repair and acceptance methods in the LNS which let the algorithm search the feature space more efficiently with improved intensification and diversification. In the following, the algorithmic flow of the WFLNS is explained. Moreover, the Pseudo-code of the WFLNS is presented at Algorithm 1, where defined functions and variables are as follows: The functions $des()$, $rep()$ and $acc()$ define destroy, repair and acceptance methods respectively. The variable xb is the best observed solution during the search, x is the current solution, xd is the outcome of the $des()$ function which would be served as the input to the $rep()$ function, and xt is the temporary solution.

3.1 Encoding Representation and Initialization

A candidate solution in WFLNS is encoded as a binary string with D digits, where D is the total number of features and we aim at choosing a string with d digits out of it, where d is the subset size. Each binary digit represents a feature values 1 and 0, in which 1 indicates a selected feature and 0 an unselected feature. As an example, string 010100 means that the second and fourth features are selected. When prior knowledge about the optimal number of features is available, we may limit d to no more than the

input : an initial solution x;
xb=x;
while *Stop criterion is not met* **do**
    xd = **des**(x) : Select K percentage of the lowest ranked selected features (ones) to remove from x ;
    xt = **rep**(xd): Select R percentage of the highest ranked features from the removed features and insert them to xt ;
    if *acc(xt , xb)* **then**
        xb = xt;
    end
end
Output : xb;

Algorithm 1: Procedure of WFLNS

predefined value; otherwise, d is equal to D. To make our results comparable with [3], we considered d equal to D in all the experiments. We initialized the initial solution at random and tried to minimize the randomness effects by repeating the experiments for five independent runs. The maximum number of iterations in [3], as our baseline for comparison, is set to the very large value , 5000, to allow all of the studied algorithms to fully converge. In our study, the number of iterations is set to 500 as in most of our experiments the solution was not improved after around 350 iterations.

3.2 Destroy and Repair Methods

The notion of neighborhood in LNS is defined by employed strategies for destructing and rebuilding the current solution to transform it to another solution. Thus both destroy and repair methods have significant impact on the quality of the final solution. Typically, the employed strategies for destroying different parts of the solution are applied randomly and the neighborhood of a solution is then defined as the set of solutions that can be reached by the repair method. The main drawback is caused by destructing the large part of the solution which leads to a neighborhood containing a large amount of candidate solutions that need to be explored. In other words, for each destruction choice there are many possible repairing solutions. In WFLNS, we incorporate a CFS filter method [45] for both destroy and repair methods. CFS is a filter method in which features are ranked based on their correlation with each other as well as the classifier. The main hypothesis behind CFS is that a good feature subsets contain features highly correlated with the classification, yet uncorrelated to each other. Thus, irrelevant features would be ignored as they typically show low correlation with the classifier. Suppose a given current solution x, we define K as a degree of destruction, which would remove K percentage of the solution (K is selected randomly and it is between 0 and 100). In order to destruct the suitable part of the solution, we employed CFS to rank features and remove lowest rank K percentage of selected features which were set to one. Afterwards, we rebuild the destructed part of the solution based on R, where R is the size of the neighborhood (R is selected randomly and it is between 1 and K). In other words, in the destroy process the lowest rank features (K percentage) would be removed and in the repair process the highest ranked features (R percentage) from the destroyed set would be considered as selected features. Given a current solution x, we define the functionality of destroy and repair methods by $des()$ and $rep()$ respectively. As mentioned before, function $des()$ selects K percentage of the lowest ranked selected features (ones) from x, using the CFS ranking selection and move it to xd. Function $rep()$ selects R percentage of the highest ranked features from destroyed set, using the CFS ranking selection and inserts them.

3.3 Objective Function

The objective function is defined by the classification accuracy, i.e.

$$ObjF(x) = Accuracy(Sx) \tag{1}$$

where Sx denotes the corresponding selected feature subset encoded in the current solution x, and the feature selection criterion function $Accuracy(Sx)$ evaluates the significance for the given feature subset Sx. As we considered paper [3] as the baseline for this work, so to make our results comparable with their results, we needed to use the same classifiers as our objective functions: a tree based classification (C4.5) [46] and the probabilistic Bayesian classifier [47] with naive independence assumptions.

3.4 Acceptance Method

We propose a new acceptance method for WFLNS, which inspired from acceptance probably of Simulated Annealing algorithm. In our proposed method xt is always accepted as xb if ObjF(xt) > ObjF(xb), and accepted with probability $e^{-(ObjF(xb)-ObjF(xt))/T}$ if $ObjF(xb) > ObjF(xt)$. Here we set the initial temperature to $T = 1$. By choosing this temperature non improving solutions would be allowed to be accepted. Within the search progress, T decreases and towards the end of the search only a few or non improving solutions would be accepted. It is worth noting that based on the original LNS in [40], the acceptance method evaluates solutions of the neighborhood and then allows the solution with the highest objective function value to be considered as a current solution if and only if it was improved. The particularity of the proposed acceptance method against its original counterpart is that it allows WFLNS move to a new solution, which might make the objective function worse in the hope of not trapping in to local optimal solutions.

4 Experimental Results and Discussion

We considered the paper [3] as a baseline for conducting and comparing our experiments because to the best of our knowledge this paper has been the most recent comparison of existing wrapper methods. We used the same classifiers, Naive Bayesian (NB) and C4.5, with the same setting to measure the classification accuracy. Then, we employed the same twelve real value data sets (Table 1) with [3], which were originally chosen from the UCI repository [4]. Data sets are both high and low dimension to present reasonable challenges to the proposed algorithm.Finally, we compared the performance of the WFLNS with all state of the art employed metaheuristic algorithms in [3], such as Swarm Intelligence algorithms, Evolutionary algorithms and Local Search algorithms. Table 1 illustrates the accuracy on all data sets for both employed classifiers with no feature selection. For the sake of performance analysis, we categorized these data sets into three main categories: small size data sets (number of features below 20), medium size data sets (number of features between 20 and 100) and large size data sets (number of features more than 100). The small sized data sets include Cleveland and Hearth. The next six data sets include Ionosphere, Water, Waveform, Ozone, Sonar and Libras are categorized as medium sized. The last four data sets Libras, Arrhythmia, Handwritten, Secom and multifeat are considered as large size data sets. The obtained results have been discussed in terms of classification accuracy, destruction degree and acceptance criteria. For each data set, the final accuracy was obtained by averaging out the accuracy of WFLNS over five independent runs with an identical initial solution. Also, the number of iterations and initial temperature (T) were set to 500 and 1 respectively.

Table 1. Data set information

Data set	Feature	Instance	Class	C4.5(%)	NB (%)
Heart	14	270	2	77.56	84.00
Cleveland	14	297	5	51.89	55.36
Ionosphere	35	230	2	86.22	83.57
Water	39	390	3	81.08	85.40
Waveform	41	699	2	75.49	79.99
Sonar	60	208	2	73.59	67.85
Ozone	73	2,534	2	92.70	67.66
Libras	91	360	15	68.24	63.635
Arrhythmia	280	452	16	65.97	61.40
Handwritten	257	1,593	10	75.74	86.21
Secom	591	1,567	2	89.56	30.04
Mutifeat	650	2000	10	94.54	95.30

4.1 Classification Accuracy

As Table 2 shows, for both classifiers the proposed method achieved the highest accuracy in comparison with other algorithms in most data sets. WFLNS achieved the highest accuracy for seven and eight out of twelve data sets for C4.5 and NB respectively. More specifically, in case of C4.5 classifier: for small size data set WFLNS achieved the highest accuracy among all other algorithms. For medium size data set (Ionosphere, Water, Waveform, Sonar and Ozone), the best subset was found by WFLNS for both Ionosphere and Sonar with 89.9 and 76.2 respectively while for Water data set GA achieved the highest accuracy with 83.4 and for Waveform data set CSA and SA achieved the same highest result with 77.6. In case of large data set, WFLNS reached the best classification accuracy for Arrhythmia data set with 68.1. For Secom and Multifeat data set, ABC and SA gain the highest accuracy with 93.4 and 95.1 respectively.

Table 2. C4.5 (Left) and NB (right) Classification Accuracies

Data set	ABC		ACO		CSA		FF		GA		HS		MA		PSO		SA		TS		WFLNS	
Heart	81.7	83.0	81.2	84.1	81.7	83.0	**82.2**	83.0	80.6	85.0	80.6	85.0	80.6	85.0	80.7	85.0	82.1	82.3	81.7	82.9	**83.2**	**85.5**
Cleveland	55.8	56.3	55.3	56.5	55.8	56.3	55.8	56.6	56.4	56.9	56.4	56.9	55.0	55.7	56.4	56.9	55.0	55.7	56.3	**57.0**	**57.8**	55.3
Ionosphere	82.2	86.6	87.6	86.3	87.8	86.8	88.4	86.9	88.4	**88.9**	88.3	**88.9**	88.3	86.9	88.0	86.4	87.4	87.0	87.3	87.0	**89.9**	88.3
Water	81.9	84.9	82.7	85.8	82.8	**85.9**	83.4	85.9	**83.4**	**85.9**	83.3	**85.9**	83.3	**85.9**	82.1	85.2	83.0	85.6	82.9	85.6	81.4	**85.9**
Waveform	76.9	79.8	77.4	80.5	**77.6**	80.7	76.8	79.5	75.5	80.2	77.5	80.2	77.5	80.2	76.9	79.7	**77.6**	86.9	77.5	80.5	76.6	**82.0**
Sonar	72.3	66.6	73.2	66.3	73.0	66.6	73.4	65.9	73.1	66.6	73.2	66.6	73.3	66.5	72.8	66.3	72.3	66.6	74.1	66.9	**76.2**	**67.3**
Ozone	93.4	75.8	93.4	77.2	93.1	74.8	93.3	76.1	93.3	73.9	93.2	73.9	93.4	73.7	93.3	73.5	93.4	78.4	93.1	74.0	**95.1**	**79.2**
Libras	65.6	60.7	62.1	57.3	67.0	61.6	65.5	61.0	67.6	62.1	67.3	61.6	68.2	61.8	66.8	61.4	65.9	61.4	66.9	61.3	**68.3**	**63.0**
Arrhythmia	63.0	63.2	66.8	67.0	67.1	68.5	66.8	67.0	66.8	66.8	66.9	68.9	66.9	67.4	63.5	63.3	67.4	69.0	67.2	69.0	**68.1**	**70.2**
Handwritten	75.0	83.7	70.2	72.9	75.9	84.7	74.3	82.0	75.5	85.2	75.9	85.3	75.4	85.5	75.3	85.3	76.0	83.8	**78.1**	84.9	77.1	**86.1**
Secom	**93.4**	88.7	92.5	82.6	92.5	84.2	92.7	75.1	90.7	84.1	92.1	71.2	91.2	**88.7**	92.7	74.2	92.5	84.7	92.4	83.7	89.5	86.7
Mutifeat	94.3	95.7	93.4	95.2	94.9	97.1	92.8	95.7	94.6	96.3	94.9	96.8	94.6	95.8	94.7	95.9	**95.1**	**97.2**	94.9	**97.2**	92.8	95.8

In case of NB classifier, WFLNS achieved the highest accuracy for Heart data set by 85.5 while for Cleveland data set, TS achieved the highest accuracy with 57.0. For all medium size data sets, the proposed algorithm outperforms other algorithms by achieving 85.9, 82.0, 67.3, 79.2 and 63 for Ionosphere,Water,Waveform, Sonar, Ozone and Libras data sets respectively. For large size data sets, apart from Secom and Multifeat data sets, WFLNS gain the highest accuracy for Arrhythmia and Handwritten with 70.2 and 86.1 classification accuracy. Moreover, Table 3 shows the number of selected features along with the selected features by WFLNS for medium size data sets, where $d(C4.5)$ and $d(NB)$ represent the number of selected features by C4.5 and NB classifiers respectively.

4.2 Effect of Destruction Degree Parameter

Choosing an appropriate degree of destruction(K), for destroy method has an impact on the quality of the search process and consequently on the quality of classification accuracy. We chose C4.5 classifier along with Ionosphere, Sonar and Libras data sets as a representative of small size, medium size and large size data set respectively to investigate the effect of destruction degree parameter on the quality of the classification accuracy.

As illustrates in Figure 1, the best classification accuracy were achieved by 0.5, 0.6 and 0.6 destruction degree in Ionosphere, Sonar and Libras data sets respectively. So, we can conclude that selecting either small or too large destruction degree lead to undesirable effects because if the small percentage of the solution is destructed then the effect of a large neighborhood search would be lost in WFLNS as it explores the smaller part of the search space and subsequently it failed to achieve the highest classification accuracy. On the other hand, if the large percentage of the solution is selected to be destructed then WFLNS turns in to random search.

Table 3. Selected Features by WFLNS

Data Set	d(C4.5)	d(NB)	Selected Features by C4.5	Selected Features by NB
Waveform	13	17	{2,4,6,7,10,11,13,15,17,18, 19,20,21}	{1,2,3,4,5,6,7,8,10,11, 12,13,14,18,19,20,21}
Sonar	32	51	{1,2,3,4,6,7,8,9,14,17,18, 19,21,22,23,24,26,28,29,34,37,38, 40,41,42,45,46,47,49,52,55,58}	{1,3,4,5,6,7,8,9,10,11,13,14,15,16, 17,18,19,22,23,24,25,26,27,28, 29,30,31,32,33,34,35,36,37,38,39, 40,41,42,43,45,48,50,51,53,54,55, 56,57,58,59,60}
Ozone	43	40	{1,4,5,6,7,8,9,12,16,17, 18,19,21,22,23,24,25,27,28,30, 32,33,37,38,39,40,41,44,45,46, 47,48,52,54,55,56,57,58,61,64, 68,72,73}	{1,2,5,7,9,10,11,12,13,15, 17,18,19,20,22,23,30,31,32,37, 38,40,41,42,45,48,49,52,55,56, 57,58,60,62,65,66,67,70,71,72}
Libras	48	71	{1,3,4,7,8,15,16,17,20,23, 24,25,26,29,32,33,34,35,37,39, 40,41,46,47,48,50,55,56,57,59, 60,61,63,64,65,67,70,71,73,76, 77,80,83,84,85,86,88,90}	{1,2,3,4,5,6,7,8,9,11, 12,13,14,15,17,18,19,20,21,22, 28,29,30,32,33,35,36,37,38,39, 40,41,42,43,45,46,47,48,49,50, 51,52,53,54,55,57,59,60,61,63, 65,66,67,68,69,71,72,73,74,75, 79,81,82,83,84,85,86,87,88,89, 90}

4.3 Effect of Acceptance Criteria

The acceptance criterion has an important role in both diversification and intensification of the search process of WFLNS. Our proposed acceptance method for WFLNS, inspired from the acceptance probability in Simulated Annealing algorithm, in which even non-improving temporary solutions would have the chance to be considered as an accepted solution in a hope of finding better solution in following iterations. Based on the acceptance method in LNS algorithm, only the best improving temporary solution would be considered as an accepted solution (best solution) for each iteration.

Table 4. Effect of Acceptance Criteria

Data set	IS-C4.5	SA-C4.5	IS-NB	SA-NB
Heart	**83.2**	**83.2**	**85.5**	**85.5**
Ionosphere	88.8	**89.9**	**89.2**	88.3
Sonar	74.1	**76.2**	66.5	**67.3**
Libras	66.2	**68.3**	62.2	**63.0**
Arrhythmia	67.3	**68.1**	68.6	**70.2**

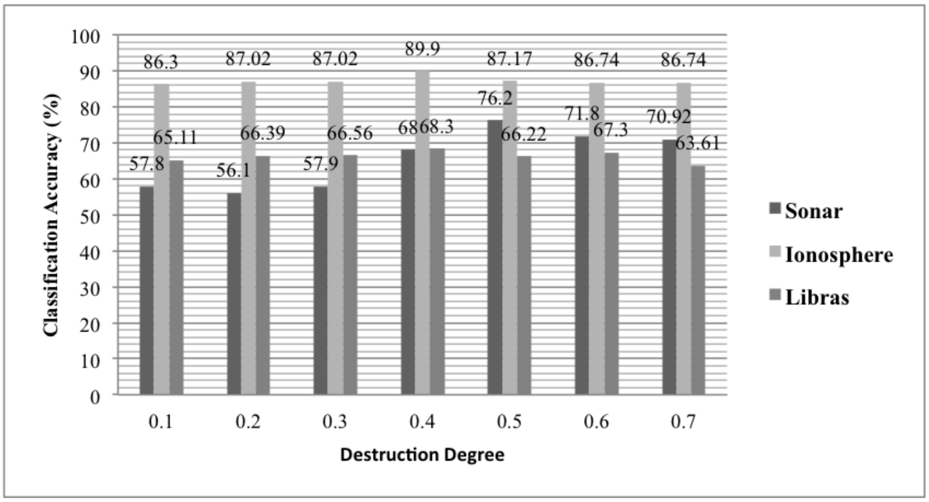

Fig. 1. Effect of destruction degree on C4.5 Classifier

To evaluate the efficiency of the proposed acceptance method, we compare the results of both methods with one another based on both classification accuracy and on different data sets size such as: Heart, Ionosphere, Sonar, Libras and Arrhythmia data sets. Two acceptance methods in Table 4 refer to as IS (Improvement strategy) and SA, which represent the LNS and WFLNS acceptance method respectively. Our experiments show that for C4.5 classifier, the proposed acceptance method achieved better results than IS, apart from the Heart data set which both methods achieved the same result. For NB classifier, the proposed method proved its superiority over IS with one exception in Ionosphere data set that achieved 89.2 by IS.

5 Conclusion

In this paper, we presented the first study on applying LNS on the problem of feature selection by proposing a new hybrid algorithm called WFLNS. The core idea of WFLNS is based on designing a problem-specific destroy, repair and acceptance methods for the LNS algorithm to deal with the problem of feature selection. We incorporated the idea of filter ranking method into the destroy and repair methods, in which the algorithm was guided into the most promising part of the search space by adding and removing proper features iteratively. Furthermore, we introduced a new acceptance method for our WFLNS, which is inspired from the Simulated Annealing acceptance probability. The particularity of the proposed acceptance method is to let both improving and non-improving solutions be considered as the best found solution.The performance of the proposed algorithm is evaluated based on C4.5 and NB classifiers and twelve real value data sets used to test the algorithms based on the paper [3], which were originally chosen from the UCI repository. Experimental results show the proposed algorithm outperforms other metaheuristic search algorithms in most data sets.

References

1. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. The Journal of Machine Learning Research **3** (2003) 1157–1182
2. Hall, M.A.: Correlation-based feature selection of discrete and numeric class machine learning. In Proceedings of the Seventeenth International Conference on Machine Learning (2000)
3. Diao, R., Shen, Q.: Nature inspired feature selection meta-heuristics. Artificial Intelligence Review **44**(3) (2015) 311–340
4. Lichman, M.: UCI machine learning repository (2013)
5. Amaldi, E., Kann, V.: On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. Theoretical Computer Science **209**(1) (1998) 237–260
6. Glover, F.: Tabu search-part i. ORSA Journal on computing **1**(3) (1989) 190–206
7. Glover, F.: Tabu searchpart ii. ORSA Journal on computing **2**(1) (1990) 4–32
8. Brooks, S.P., Morgan, B.J.: Optimization using simulated annealing. The Statistician (1995) 241–257
9. Geem, Z.W., Kim, J.H., Loganathan, G.: A new heuristic optimization algorithm: harmony search. Simulation **76**(2) (2001) 60–68
10. Wang, Y., Li, L., Ni, J., Huang, S.: Feature selection using tabu search with long-term memories and probabilistic neural networks. Pattern recognition letters **30**(7) (2009) 661–670
11. Wang, Y., Mo, J.: Emotion feature selection from physiological signals using tabu search. In: 2013 25th Chinese Control and Decision Conference (CCDC). (2013)
12. Lin, S.W., Lee, Z.J., Chen, S.C., Tseng, T.Y.: Parameter determination of support vector machine and feature selection using simulated annealing approach. Applied soft computing **8**(4) (2008) 1505–1512
13. Meiri, R., Zahavi, J.: Using simulated annealing to optimize the feature selection problem in marketing applications. European Journal of Operational Research **171**(3) (2006) 842–858
14. Miller, A.: Subset selection in regression. CRC Press (2002)
15. Gheyas, I.A., Smith, L.S.: Feature subset selection in large dimensionality domains. Pattern recognition **43**(1) (2010) 5–13
16. Wang, Y., Liu, Y., Feng, L., Zhu, X.: Novel feature selection method based on harmony search for email classification. Knowledge-Based Systems **73** (2015) 311–323
17. Shreem, S.S., Abdullah, S., Nazri, M.Z.A.: Hybrid feature selection algorithm using symmetrical uncertainty and a harmony search algorithm. International Journal of Systems Science **47**(6) (2016) 1312–1329
18. Whitley, D.: A genetic algorithm tutorial. Statistics and computing **4**(2) (1994) 65–85
19. Moscato, P., et al.: On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Caltech concurrent computation program, C3P Report **826** (1989) 1989
20. Gao, X.Z.: Artificial immune systems and their applications. NICSO2006 (2006) 7
21. Eberhart, R.C., Kennedy, J., et al.: A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science. Volume 1., New York, NY (1995) 39–43
22. Colorni, A., Dorigo, M., Maniezzo, V., et al.: Distributed optimization by ant colonies. In: Proceedings of the first European conference on artificial life. Volume 142., Paris, France (1991) 134–142
23. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical report, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department (2005)

24. Oh, I.S., Lee, J.S., Moon, B.R.: Hybrid genetic algorithms for feature selection. Pattern Analysis and Machine Intelligence, IEEE Transactions on **26**(11) (2004) 1424–1437
25. Ghamisi, P., Benediktsson, J.A.: Feature selection based on hybridization of genetic algorithm and particle swarm optimization. IEEE Geoscience and Remote Sensing Letters **12**(2) (2015) 309–313
26. Lee, J., Kim, D.W.: Memetic feature selection algorithm for multi-label classification. Information Sciences **293** (2015) 80–96
27. Zhu, Z., Ong, Y.S., Dash, M.: Wrapper–filter feature selection algorithm using a memetic framework. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on **37**(1) (2007) 70–76
28. Shojaie, S., Moradi, M.: An evolutionary artificial immune system for feature selection and parameters optimization of support vector machines for erp assessment in a p300-based gkt. In: Biomedical Engineering Conference, 2008. CIBEC 2008. Cairo International, IEEE (2008) 1–5
29. Kashef, S., Nezamabadi-pour, H.: An advanced aco algorithm for feature subset selection. Neurocomputing **147** (2015) 271–279
30. Forsati, R., Moayedikia, A., Jensen, R., Shamsfard, M., Meybodi, M.R.: Enriched ant colony optimization and its application in feature selection. Neurocomputing **142** (2014) 354–371
31. Wang, X., Yang, J., Teng, X., Xia, W., Jensen, R.: Feature selection based on rough sets and particle swarm optimization. Pattern Recognition Letters **28**(4) (2007) 459–471
32. Hancer, E., Xue, B., Zhang, M., Karaboga, D., Akay, B.: A multi-objective artificial bee colony approach to feature selection using fuzzy mutual information. In: 2015 IEEE Congress on Evolutionary Computation (CEC), IEEE (2015) 2420–2427
33. Yang, C.H., Chuang, L.Y., Yang, C.H., et al.: Ig-ga: a hybrid filter/wrapper method for feature selection of microarray data. Journal of Medical and Biological Engineering **30**(1) (2010) 23–28
34. Hu, Z., Bao, Y., Xiong, T., Chiong, R.: Hybrid filter–wrapper feature selection for short-term load forecasting. Engineering Applications of Artificial Intelligence **40** (2015) 17–27
35. Fahy, C., Ahmadi, S., Casey, A.: A comparative analysis of ranking methods in a hybrid filter-wrapper model for feature selection in dna microarrays. In: Research and Development in Intelligent Systems XXXII. Springer (2015) 387–392
36. Swiniarski, R.W., Skowron, A.: Rough set methods in feature selection and recognition. Pattern recognition letters **24** (2003) 833–849
37. Bermejo, P., Gámez, J.A., Puerta, J.M.: A grasp algorithm for fast hybrid (filter-wrapper) feature subset selection in high-dimensional datasets. Pattern Recognition Letters **32**(5) (2011) 701–711
38. Bertolazzi, P., Felici, G., Festa, P., Lancia, G.: Logic classification and feature selection for biomedical data. Computers and Mathematics with Applications **55**(5) (2008) 889–899
39. Bertolazzi, P., Felici, G., Festa, P., Fiscon, G., Weitschek, E.: Integer programming models for feature selection: New extensions and a randomized solution algorithm. European Journal of Operational Research **250**(2) (2016) 389–399
40. Shaw, P.: Using constraint programming and local search methods to solve vehicle routing problems. In: Principles and Practice of Constraint ProgrammingCP98. Springer (1998) 417–431
41. Pisinger, D., Ropke, S.: Large neighborhood search. In: Handbook of metaheuristics. Springer (2010) 399–419
42. Lin, S., Kernighan, B.W.: An effective heuristic algorithm for the traveling-salesman problem. Operations research **21**(2) (1973) 498–516
43. Demirovic, E., Musliu, N.: Maxsat based large neighborhood search for high school timetabling. Computers and Operations Research **78** (2017) 172 – 180

44. Meyers, C., Orlin, J.B.: Very large-scale neighborhood search techniques in timetabling problems. In: Practice and Theory of Automated Timetabling VI. Springer (2006) 24–39
45. Hall, M.A.: Correlation-based feature selection for machine learning. PhD thesis, The University of Waikato (1999)
46. Quinlan, J.R.: C4. 5: programs for machine learning. Elsevier (2014)
47. John, G.H., Langley, P.: Estimating continuous distributions in bayesian classifiers. In: Proceedings of the Eleventh conference on Uncertainty in artificial intelligence, Morgan Kaufmann Publishers Inc. (1995) 338–345

Appendix 1.

Dataset	Selected Features by WFLNS on Naïve Bayes (five independent runs)	Accuracy (NB)
Heart	[0,1,1,0,1,1,1,1,1,1,1,1,1]	85.92
	[0,1,1,1,1,1,0,1,1,1,1,1,1]	86.29
	[0,1,1,0,0,0,1,1,1,1,1,1,1]	85.55
	[0,1,1,1,1,1,0,1,1,1,1,1,1]	86.29
	[0,1,1,0,1,1,1,1,1,1,1,1,1]	85.92
Cleveland	[1,0,1,1,1,0,1,1,1,0,1,1,1]	55.50
	[1,0,1,1,1,0,1,1,1,0,1,1,1]	54.21
	[0,0,1,0,0,1,1,1,1,0,1,0,0]	57.54
	[0,0,1,1,0,0,0,1,1,1,1,0,1]	57.88
	[1,0,1,1,1,0,1,1,1,0,1,1,1]	58.24
Ionosphere	[1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,0,1,0,0,1,1]	91.93
	[0,1,0,0,1,1,0,1,0,0,1,1,0,1,0,1,0,0,1,0,0,0,0,0,1,0,0,0,1,0,0,0,1,0]	90.80
	[0,1,0,0,1,1,0,1,0,1,1,1,1,1,0,1,0,0,1,0,0,0,1,0,1,0,0,0,1,0,0,0,1,0]	91.93
	[0,1,0,0,1,1,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,1,0,0,0,0,1,0]	87.46
	[1,0,0,1,1,1,0,1,0,1,1,1,1,1,1,1,1,1,0,1,1,1,0,1,0,1,1,1,0,1,0,1,1]	88.89
Water	[0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0,0,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,0,1]	86.01
	[0,0,0,0,1,1,1,1,1,1,1,0,1,0,0,1,0,0,0,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,0,1]	85.22
	[0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0,0,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,0,1]	86.21
	[0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0,0,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,0,1]	86.01
	[0,0,0,0,1,1,1,1,1,1,1,0,0,0,1,1,0,0,0,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,0,1]	85.43
	[1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1]	80.74
	[1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1]	80.74

Waveform	[1,1,1,1,0,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,0,1,0,1,0,1,1,1,1,1,0,1,1,1,1,1,1,1,1]	82.10
	[1,1,1,1,0,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,0,1,0,1,0,1,1,1,1,1,0,1,1,1,1,1,1,1,1]	82.10
	[1,1,1,1,0,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,0,1,0,1,0,1,1,1,1,1,0,1,1,1,1,1,1,1,1]	82.62
Sonar	[0,1,0,1,0,1,1,1,0,0,1,1,0,1,0,1,0,0,1,0,0,1,1,0,0,1,0,1,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0, 0,0,0,0,1,1,0,1,1,0,1,1]	67.42
	[1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,0,0,1, 1,1,0,1,1,1,1,1,0,1,1,1]	68.40
	[1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,0,0,0,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,0,0,1, 1,1,0,1,1,1,1,1,0,1,1,1]	67.92
	[1,1,1,1,1,1,1,1,0,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1, 0,1,1,1,1,1,1,1,1,1,1,1]	68.90
	[1,1,1,1,1,1,1,1,0,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1, 0,1,1,1,1,1,1,1,1,1,1,1]	68.90
Ozone	[0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,1,0,0,0,0, 0,0,1,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0]	79.91
	[0,0,1,0,0,0,0,0,0,0,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,0,0, 0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]	82.0
	[0,1,0,0,0,0,0,0,0,0,1,0,1,1,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,0,1,1,1,1,1,0,0,0,0, 0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0]	79.23
	[0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0, 0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]	80.58
	[0,0,1,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0, 0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,1,1,1,1,0]	79.01
Libras	[0,1,1,0,1,1,1,1,1,0,0,0,0,0,1,0,1,1,1,0,1,0,0,1,1,0,1,0,1,0,1,1,1,0,1,0,1,1,0,0,1,1,1,1,0,0,0,0, 0,0,1,1,1,1,1,0,0,1,0,0,1,0,1,0,1,0,1,1,0,0,0,1,1,0,1,1,1,0,1,0,1,1,0,0,0,1,0,1,0,0]	63.61
	[1,0,1,0,1,0,1,1,0,0,0,1,1,1,0,1,1,1,1,0,1,0,0,1,0,0,1,1,1,1,0,1,0,1,0,1,0,1,0,1,1,0,1,1,1,0,0,1, 1,1,1,1,0,0,1,1,1,0,1,1,1,0,0,0,0,0,0,0,1,1,1,0,0,0,1,0,1,0,1,0,1,1,1,1,0,0,1,1,1]	67.50
	[0,1,1,0,1,1,0,0,1,0,0,1,1,1,0,1,1,0,1,1,0,1,1,0,1,1,1,1,0,0,1,0,1,0,0,0,0,1,0,0,1,1,1,1,0,1,1,1, 1,1,1,0,0,1,0,0,1,0,1,1,1,0,0,0,1,1,0,0,1,1,0,0,1,1,1,0,1,1,0,0,1,1,0,1,1,0,1,1,1,1]	64.16
	[0,0,1,0,0,0,1,0,1,1,1,0,0,0,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,1,0,0,0,1,1,0,1,0,1,0,1,	62.22

	0,0,1,0,0,0,0,1,1,0,0,1,0,1,1,0,0,1,0,0,0,1,1,1,0,0,1,1,1,1,1,0,1,0,0,0,1,0,0,0,0,1]	
	[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,0,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]	64.44
Arrhythmia	[0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,1,0,0,0, 1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,1,1,0,0,1,1,0,0,0,0,0,0,1,0,0,1,0,0,1,0,0,0,1,0,0,0,0,1,0,1,0, 0,0,0,1,0,1,1,0,1,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,1,1,0,1,0,0,1,0,0,1, 0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,1,0,1,0,1,0,1,0,1,1,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,1,1, 0,1,0,0,0,1,0,0,0,1,1,0,0,0,1,0,1,0,0,0,0,0,0,0,1,1,0,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0, 0,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,1,1,1,1,0,0,1,0,0,0,1,0,1,1,0,0,1,0,1,0,1,0,0]	69.40
	[0,1,0,0,1,0,0,1,1,0,0,0,0,0,1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0, 1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,1,0,0,1,0,0,0,0,0,0,1,0,1,0,0,1,0,0,1,0,0,0,1,0,1,0,0,1,0,1,0, 0,0,0,1,0,1,1,0,1,0,1,0,1,1,0,1,1,0,0,0,0,0,0,1,0,1,0,1,1,0,0,0,0,0,0,0,1,1,0,1,0,1,1,0,0,1, 0,0,0,1,0,0,0,1,1,0,0,0,0,1,0,1,0,0,1,0,1,0,1,0,1,1,1,1,0,0,0,0,0,1,0,0,1,0,1,0,0,0,0,0,0,1,1,1, 0,1,0,0,1,1,1,0,0,1,1,0,0,0,1,0,1,0,1,0,1,0,0,0,0,1,1,0,1,1,0,1,0,0,1,0,0,0,1,0,0,0,0,0,1,1,0,0, 1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,1,0,1,1,1,0,1,0,1,0,0,1,0,1,0,0,1,0,1,0,0,0,1]	69.27
	[0,1,0,0,1,0,0,1,1,0,0,0,1,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1, 0,0,0,0,0,0,0,0,0,1,0,0,0,0,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,1,1,0,1,1,0,0,0,0,0,0,1,0, 0,1,0,0,0,0,1,0,0,0,1,0,1,1,0,0,1,0,1,0,1,0,1,0,1,0,0,0,0,0,1,1,0,0,1,0,0,0,1,0,1,1,1,0,1,0,0,0, 0,0,1,1,0,0,0,1,1,0,0,0,0,1,0,1,0,1,0,1,1,0,0,1,0,0,1,1,1,1,0,0,0,1,0,0,0,0,0,0,0,1,1,0,0,0,0,1, 0,1,1,0,0,0,0,1,0,1,0,0,0,1,0,1,0,0,0,1,0,0,1,0,0,0,1,0,0,0,0,1,0,1,0,0,1,0,0,1,0,1,0,1,0,0,0,0, 0,1,1,1,0,0,0,1,0,0,0,0,1,1,0,1,0,0,0,1,0,1,1,1,0,1,0,0,0,0,0,1,0,1]	70.32
	[0,1,0,0,1,0,0,1,1,0,0,0,0,0,1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0, 1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,1,0,0,1,0,0,0,0,0,0,1,0,1,0,0,1,0,0,1,0,0,0,1,0,1,0,0,1,0,1,0, 0,0,0,1,0,1,1,0,1,0,1,0,1,1,0,1,1,0,0,0,0,0,0,1,0,1,0,1,1,0,0,0,0,0,0,0,1,1,0,1,0,1,1,0,0,1, 0,0,0,1,0,0,0,1,1,0,0,0,0,1,0,1,0,0,1,0,1,0,1,0,1,1,1,1,0,0,0,0,0,1,0,0,1,0,1,0,0,0,0,0,0,1,1,1, 0,1,0,0,1,1,1,0,0,1,1,0,0,0,1,0,1,0,1,0,1,0,0,0,0,1,1,0,1,1,0,1,0,0,1,0,0,0,1,0,0,0,0,0,1,1,0,0, 1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,1,0,1,1,1,0,1,0,1,0,0,1,0,1,0,0,1,0,1,0,0,0,1]	71.21
	[0,1,0,0,1,0,0,1,1,0,0,0,0,0,1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0, 1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,1,0,0,1,0,0,0,0,0,0,1,0,1,0,0,1,0,0,1,0,0,0,1,0,1,0,0,1,0,1,0, 0,0,0,1,0,1,1,0,1,0,1,0,1,1,0,1,1,0,0,0,0,0,0,1,0,1,0,1,1,0,0,0,0,0,0,0,1,1,0,1,0,1,1,0,0,1,	71.21

<table>
<tr><td></td><td>0,0,0,1,0,0,0,1,1,0,0,0,0,1,0,1,0,0,1,0,1,0,1,0,1,1,1,1,0,0,0,0,0,1,0,0,1,0,1,0,0,0,0,0,0,1,1,1,
0,1,0,0,1,1,1,0,0,1,1,0,0,0,1,0,1,0,1,0,1,0,0,0,0,1,1,0,1,1,0,1,0,0,1,0,0,0,1,0,0,0,0,0,1,1,0,0,
1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,1,0,1,1,1,0,1,0,1,0,0,1,0,1,0,0,1,0,1,0,0,0,1]</td><td></td></tr>
<tr><td rowspan="5">Handwritten</td><td>[1,1,1,1,1,0,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,0,0,1,0,1,1,1,1,0,0,0,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,0,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,0,0,0,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,0,1,1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,
0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]</td><td>86.55</td></tr>
<tr><td>[1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,0,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]</td><td>85.23</td></tr>
<tr><td>[1,1,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0,1,1,1,1,1,0,0,1,1,1,1,1,1,0,1,1,1,1,1,0,1,1,1,0,1,1,1,1,
0,1,1,1,1,1,0,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,
1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,0,0,1,1,1,1,1,0,1,1,0,1,0,1,0,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,
0,1,1,1,1,1,0,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,1,1,0,1,1,0,1,1,
1,1,0,1,1,1,1,1,0,0,1,1,1,1,1,1]</td><td>86.11</td></tr>
<tr><td>[1,1,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0,1,1,1,1,1,0,0,1,1,1,0,1,1,0,1,1,1,1,1,0,1,1,1,0,1,1,1,1,
0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,
1,1,1,1,1,1,0,0,1,1,1,1,1,0,1,0,0,1,1,1,1,1,0,1,1,0,1,0,1,0,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,
0,1,1,1,1,1,0,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,0,1,
1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,1,1,0,1,1,0,1,1,
1,1,0,1,1,1,1,1,0,0,1,1,1,1,1,1]</td><td>86.42</td></tr>
<tr><td>[1,1,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0,1,1,1,1,1,0,0,1,1,1,0,1,1,0,1,1,1,1,1,0,1,1,1,0,1,1,1,1,
0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,
1,1,1,1,1,1,0,0,1,1,1,1,1,0,1,0,0,1,1,1,1,1,0,1,1,0,1,0,1,0,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,</td><td>86.42</td></tr>
</table>

	0,1,1,1,1,1,0,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,0,1, 1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,1,1,0,1,1,0,1,1, 1,1,0,1,1,1,1,1,0,0,1,1,1,1,1,1]	
Secom	[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,0,0,0,0,1,0,0,0,0,1,1,0,1,0,1,1,1,0,0,1,0,0,1,1,1, 0,0,1,1,0,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,1,0,0,1,1,0,1,0,0,0,0,0,1,0,1,1,1,1,1,0,1,1,0,1,1,1,0, 1,1,0,0,1,1,0,0,0,1,0,1,0,1,0,1,0,1,1,0,0,1,0,0,0,0,0,1,0,1,1,0,0,1,0,1,1,1,0,1,0,0,0,1,1,1,1,0, 1,1,0,0,0,1,1,0,1,1,1,0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,1,0,1,0,1,1,0,1,0,0,0,1,1,0,0,0,1,1,0,1,0, 0,1,0,0,0,0,0,0,0,0,1,0,1,1,1,0,0,0,0,1,0,1,1,1,0,0,1,0,1,0,1,1,0,0,0,0,1,1,0,0,0,1,1,1,1,0,0,1, 0,0,1,0,1,0,0,1,0,1,0,1,1,0,0,0,1,0,0,0,1,0,1,0,0,1,0,1,0,0,1,1,1,0,0,1,1,0,0,1,1,1,0,0,1,0,1,0, 0,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,1,1,1,0,1,0,0,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,1,1,0,0,0,0,0,1,0,0, 1,0,0,0,0,1,1,0,0,0,1,1,0,0,1,1,1,0,1,1,0,0,1,0,0,1,1,1,1,1,0,0,0,0,1,0,0,0,1,1,1,1,0,0,0,0,0,0, 0,0,1,0,1,1,0,0,0,0,0,0,0,1]	88.37
	[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,0,0,0,0,1,0,0,0,0,1,1,0,1,0,1,1,1,0,0,1,0,0,1,1,1, 0,0,1,1,0,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,1,0,0,1,1,0,1,0,0,0,0,0,1,0,1,1,1,1,1,0,1,1,0,1,1,1,0, 1,1,0,0,1,1,0,0,0,1,0,1,0,1,0,1,0,1,1,0,0,1,0,0,0,0,0,1,0,1,1,0,0,1,0,1,1,1,0,1,0,0,0,1,1,1,1,0, 1,1,0,0,0,1,1,0,1,1,1,0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,1,0,1,0,1,1,0,1,0,0,0,1,1,0,0,0,1,1,0,1,0, 0,1,0,0,0,0,0,0,0,0,1,0,1,1,1,0,0,0,0,1,0,1,1,1,0,0,1,0,1,0,1,1,0,0,0,0,1,1,0,0,0,1,1,1,1,0,0,1, 0,0,1,0,1,0,0,1,0,1,0,1,1,0,0,0,1,0,0,0,1,0,1,0,0,1,0,1,0,0,1,1,1,0,0,1,1,0,0,1,1,1,0,0,1,0,1,0, 0,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,1,1,1,0,1,0,0,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,1,1,0,0,0,0,0,1,0,0, 1,0,0,0,0,1,1,0,0,0,1,1,0,0,1,1,1,0,1,1,0,0,1,0,0,1,1,1,1,1,0,0,0,0,1,0,0,0,1,1,1,1,0,0,0,0,0,0, 0,0,1,0,1,1,0,0,0,0,0,0,0,1]	87.99

	[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,0,1,0,1,1,1,0,0,0,0,1,0,0,1,1,0,1,0,0,1,1,1,1,1,0,0,1,1,0,1,0,1,0,1,1, 1,0,1,1,0,0,0,1,0,0,1,0,0,0,0,1,0,1,0,0,1,1,0,0,1,0,0,1,1,1,0,1,0,1,0,1,0,0,0,0,1,0,0,1,1,1,1,0, 1,0,0,0,0,1,0,1,0,1,1,1,1,1,0,1,0,0,1,0,1,0,0,0,1,0,1,1,1,0,0,0,0,0,1,0,0,0,1,1,0,1,0,0,1,1,1,0, 1,1,0,1,1,0,1,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,1,0,1,0,1,0,1,1,0,0,1,0,0,0,1,0,0,1,1,0,1,0,0,1,0, 1,0,0,0,0,1,1,0,1,1,1,1,0,0,1,1,0,1,0,0,0,0,1,0,1,1,0,0,0,1,1,1,0,0,0,0,1,1,0,0,1,0,0,1,0,1,0,0, 1,0,0,1,1,1,1,1,0,1,0,0,1,1,0,1,1,0,0,0,1,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,1,0,1,1,0,0,0,0,0,0,0,1, 1,1,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,1,0,1,1,0,0,0,1,1,1,0,0,0,1,0,1,1,1,1,1,0,1,0,0,0, 0,1,0,0,1,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,0,0,0,0,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,0,1,1,0,1,1,0, 0,0,0,0,1,1,0,1,1,1,1,0,0,0,0,0,1,0,1,0,0,1,0,0,1,1,0,0,0,0,0,0,1,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0, 1,0,1,1,1,0,0,0,0,0,0,0,1,0,1,1,0,0,1,0,0,1,0,0,0,1,0,1,1,1,0,1,0,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0, 0,0,0,0,0,1,0,0,0,1,1,1,0,0];	86.3
	[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,0,0,0,1,0,0,0,1,1,0,1,0,1,1,1,0,0,1,0,0,1,1,1, 0,0,1,1,0,0,0,1,0,0,0,0,0,0,0,1,1,0,0,0,1,0,0,1,1,0,1,0,0,0,0,0,1,0,1,1,1,1,1,0,1,1,0,1,1,1,0, 1,1,0,0,1,1,0,0,0,1,0,1,0,1,0,1,0,1,1,0,0,1,0,0,0,0,0,1,0,1,1,0,0,1,0,1,1,1,0,1,0,0,0,1,1,1,1,0, 1,1,0,0,0,1,1,0,1,1,1,0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,1,0,1,0,1,1,0,1,0,0,0,1,1,0,0,0,1,1,0,1,0, 0,1,0,0,0,0,0,0,0,0,1,0,1,1,1,0,0,0,0,1,0,1,1,1,0,0,1,0,1,0,1,1,0,0,0,0,1,1,0,0,0,1,1,1,1,0,0,1, 0,0,1,0,1,0,0,1,0,1,0,1,1,0,0,0,1,0,0,0,1,0,1,0,0,1,0,1,0,0,1,1,1,0,0,1,1,0,0,1,1,1,0,0,1,0,1,0, 0,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,1,1,1,0,1,0,0,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,1,1,0,0,0,0,0,1,0,0, 1,0,0,0,0,1,1,0,0,0,1,1,0,0,1,1,1,0,1,1,0,0,1,0,0,1,1,1,1,1,0,0,0,0,1,0,0,0,1,1,1,1,0,0,0,0,0,0, 0,0,1,0,1,1,0,0,0,0,0,0,0,1]	87.3
	[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,	86.8

	1,1,1,1,1,1,1,1,1,1,1,0,1,0,1,1,1,0,0,0,0,0,1,0,0,0,1,1,0,1,0,0,1,1,1,1,1,0,0,1,1,0,1,0,1,0,1,1, 1,0,1,1,0,0,0,1,0,0,1,0,0,0,0,1,0,1,0,0,1,1,0,0,1,0,0,1,1,1,0,1,0,1,0,1,0,0,0,0,1,0,0,1,1,1,1,0, 1,0,0,0,0,1,0,1,0,1,1,1,1,1,0,1,0,0,1,0,1,0,0,0,1,0,1,1,1,0,0,0,0,0,1,0,0,0,1,1,0,1,0,0,1,1,1,0, 1,1,0,1,1,0,1,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,1,0,1,0,1,0,1,1,0,0,1,0,0,0,1,0,0,1,1,0,1,0,0,1,0, 1,0,0,0,0,1,1,0,1,1,1,1,0,0,1,1,0,1,0,0,0,0,1,0,1,1,0,0,0,1,1,1,0,0,0,0,1,1,0,0,1,0,0,1,0,1,0,0, 1,0,0,1,1,1,1,1,0,1,0,0,1,1,0,1,1,0,0,0,1,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,1,0,1,1,0,0,0,0,0,0,0,1, 1,1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,1,0,1,1,0,0,0,1,1,1,0,0,0,1,0,1,1,1,1,1,0,1,0,0,0, 0,1,0,0,1,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,0,0,0,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,0,1,1,0,1,1,0, 0,0,0,0,1,1,0,1,1,1,1,0,0,0,0,0,1,0,1,0,0,1,0,0,1,1,0,0,0,0,0,0,1,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0, 1,0,1,1,1,0,0,0,0,0,0,1,0,1,1,0,0,1,0,0,1,0,0,0,1,0,1,1,1,0,1,0,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0, 0,0,0,0,0,1,0,0,0,1,1,1,0,0]	
Multifeat	[1,0,0,1,1,1,0,0,0,1,1,1,1,0,0,0,0,0,1,0,1,0,0,0,1,1,0,1,0,0,0,0,0,1,0,1,1,1,1,0,1,1,0,0,1,0, 1,0,1,1,0,1,1,0,1,1,0,0,1,0,0,0,1,0,0,1,1,1,0,0,1,1,0,0,1,1,1,1,0,1,1,1,0,1,1,0,1,0,0,1,0,0,0,1, 0,1,0,0,1,0,0,1,0,0,1,0,1,0,0,1,0,0,1,1,0,1,0,0,1,0,1,1,1,0,0,0,0,1,0,1,1,1,0,1,1,0,1,1,1,0,0,0, 0,1,0,1,0,1,0,1,1,1,1,0,1,0,1,1,0,1,0,0,0,1,0,1,0,0,1,1,0,1,0,0,0,1,0,1,0,0,1,1,0,0,1,1,1,1,0,0, 0,0,1,1,0,1,0,1,0,0,0,1,1,0,0,1,0,0,1,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,1,1,1,0,1,0,1,1,1,1,0,1,0,1, 0,0,0,1,1,0,1,1,0,1,1,1,1,0,0,0,1,0,0,1,1,0,1,0,0,0,1,0,0,1,1,1,0,1,1,1,0,0,0,0,0,1,0,1,0,0,1,1, 0,1,1,1,0,1,1,1,0,0,0,0,0,0,1,1,1,1,0,0,1,1,1,1,1,0,1,1,0,0,0,0,0,0,0,1,1,0,1,1,0,0,1,1,0,1,1,0, 0,1,1,0,0,0,0,0,1,1,0,1,0,0,0,0,1,1,0,0,0,0,0,1,0,1,1,0,0,0,0,0,1,1,0,0,0,0,1,1,1,0,1,0,0,1,0,0, 0,1,0,1,0,0,0,0,0,1,1,0,1,0,1,1,1,0,1,0,1,0,1,1,1,1,1,0,0,1,1,1,1,1,0,0,0,1,1,0,1,1,1,0,1,1,1,0, 1,1,0,0,1,1,1,1,1,1,1,1,0,1,1,0,0,0,1,1,1,0,1,0,0,0,1,1,0,0,1,1,0,0,1,0,1,1,1,0,0,1,0,1,0,0,0,0, 1,0,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,1,0,0,0,1,0,1,1,0,1,0,1,1,0,1,0,0,0,0,1,0,1,0,1,1,1,0,0,0,1,0, 0,0,1,0,1,0,0,0,0,0,0,1,0,0,0,1,1,1,1,0,1,1,0,1,0,1,1,1,0,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,1,0,1, 0,1,1,1,1,0,1,0,1,1,1,1,0,1,0,1,0,1,0,0,0,1,0,1,0,0,1,1,1,0,0,1,0,1,1,1,0,0,0,1,0,1,1,0,1,0,0,0, 1,1,0,0,1,1,1,1,1,0,1,0,1,0,0,0,0,1,1,0,1,0,1,0,1]	96.04
	[1,0,1,1,1,1,1,1,1,0,0,0,0,1,1,0,0,0,1,1,0,1,0,0,0,1,1,0,1,0,0,1,0,0,0,1,0,0,1,0,0,1,1,1,0,1,0,0, 0,0,0,1,0,1,0,1,1,1,0,0,1,1,1,0,0,0,1,1,1,1,1,0,0,1,1,0,0,1,0,1,0,1,0,1,1,0,1,0,0,1,0,0,0,0,1,0, 0,1,0,0,1,0,1,1,1,1,0,1,1,1,1,1,1,0,1,1,0,1,1,1,1,0,1,0,1,0,0,0,0,0,1,0,0,0,1,1,0,0,0,0,1,1,1, 1,1,0,0,0,0,1,0,0,1,1,1,0,0,0,1,0,1,1,1,1,1,0,1,1,1,0,0,1,0,0,0,0,0,1,1,0,1,1,1,0,0,1,0,1,1,0,0, 0,0,0,1,1,1,0,0,1,1,0,0,0,0,1,0,1,1,0,0,0,0,0,1,0,1,1,1,1,1,1,0,0,1,1,1,0,0,1,1,0,0,0,0,0,1,1,0,	96.55

	0,0,0,1,0,1,1,1,0,1,0,0,1,0,1,1,1,0,0,1,0,1,1,0,1,0,0,0,0,1,1,0,1,1,0,1,0,1,0,0,1,0,1,1,1,1,0, 0,1,1,0,1,1,1,1,1,0,1,0,0,1,0,0,0,1,1,1,1,1,0,0,1,1,0,1,1,1,1,1,1,1,1,0,0,0,0,1,0,1,0,1,1, 1,0,1,1,0,1,1,1,0,0,1,1,0,0,1,1,1,0,0,1,1,0,1,0,0,1,1,1,1,1,1,1,0,0,0,1,1,0,0,1,0,0,0,0,1,1,1,1, 1,1,1,1,1,1,1,0,1,0,0,1,1,0,1,1,0,0,0,0,0,0,1,0,1,0,0,0,0,1,1,1,0,1,1,0,1,1,1,1,1,1,0,0,0,1,1, 1,1,0,1,1,1,1,0,0,1,0,0,1,0,1,1,0,0,1,0,0,1,1,0,0,1,0,0,1,1,0,0,0,0,0,1,1,0,1,0,0,1,1,1,1,1,0, 0,0,0,1,0,1,0,1,0,0,0,1,1,0,1,1,1,0,1,1,1,1,1,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,0,0,1,0,0,0, 1,0,1,0,1,0,1,1,1,0,1,1,1,1,0,1,1,1,0,1,1,0,1,0,0,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0, 0,0,0,1,1,0,1,0,1,0,1,0,1,0,0,1,0,0,0,0,1,0,1,0,1,1,1,1,1,1,1,1,1,0,1,0,1,0,1,1,1,0,1,0,0,0,0,0, 1,0,1,1,1,0,0,0,1,0,0,1,1,0,0,0,0,1,0,1,1,1,0,0,0]	
	[1,0,1,1,1,1,1,1,1,0,0,0,0,1,1,0,0,0,1,1,0,1,0,0,0,1,1,0,1,1,0,1,0,0,0,1,0,0,1,0,0,1,1,1,0,1,0,0, 0,0,0,1,0,1,0,1,1,1,0,0,1,1,1,0,0,0,1,1,1,1,1,0,0,1,1,0,0,1,0,1,0,1,0,1,1,0,1,0,0,1,0,0,0,0,1,0, 0,1,0,0,1,0,1,1,1,1,0,1,1,1,1,1,1,0,1,1,0,1,0,1,1,0,1,0,1,0,0,0,0,0,1,0,0,0,1,1,0,0,0,0,1,1,1, 1,1,0,0,0,0,1,0,0,1,1,0,0,0,0,1,0,1,1,1,1,1,0,1,1,1,0,0,1,0,0,0,0,0,1,1,0,1,1,1,0,0,1,0,1,1,0,0, 0,0,0,1,1,1,0,0,1,1,0,0,0,0,1,0,1,1,0,0,0,0,0,1,0,1,1,1,1,1,1,0,0,1,1,1,0,0,1,1,0,0,0,0,0,1,1,0, 0,0,0,1,1,1,1,1,0,0,0,0,1,0,1,1,1,0,0,1,0,1,1,0,1,0,0,0,0,0,1,1,0,1,1,0,1,0,1,0,0,1,0,1,1,1,1,0, 0,1,1,0,1,1,1,1,1,0,1,0,0,1,0,0,0,0,1,1,1,1,1,0,0,1,1,0,0,1,1,1,1,1,1,1,1,0,0,0,0,0,1,0,1,0,1,1, 1,0,1,1,0,1,1,1,0,0,1,0,0,0,1,1,1,0,0,0,1,0,1,0,0,1,1,1,1,1,0,1,0,0,0,1,1,0,0,1,0,0,0,0,1,1,1,1, 1,1,1,1,1,1,1,0,1,0,0,1,1,0,1,1,0,0,0,0,0,0,1,0,1,0,0,0,0,1,1,1,0,1,1,0,0,1,1,1,1,1,0,0,0,1,1, 1,1,0,1,1,1,1,0,0,0,0,0,1,0,1,1,0,0,1,0,0,1,1,0,0,1,0,0,1,1,1,0,0,0,0,0,1,1,0,0,0,0,1,1,1,1,1,0, 0,0,0,1,0,1,0,1,0,0,0,1,1,0,1,1,1,0,1,1,1,1,1,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,0,0,1,0,0,0, 1,0,1,0,1,0,1,1,1,0,1,1,1,1,0,1,1,1,0,1,1,0,1,0,0,1,1,1,0,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0, 0,0,0,1,1,0,1,0,0,0,1,0,1,0,0,1,0,0,0,0,1,1,1,0,1,1,1,1,1,1,1,1,1,0,1,0,1,0,1,1,1,0,1,0,0,0,0,0, 1,0,1,1,1,0,0,0,1,0,0,1,1,0,0,0,0,1,0,1,1,1,0,0,0];	96.60
	[1,0,0,1,0,1,1,0,0,0,0,1,0,1,1,1,1,1,1,1,1,1,0,1,0,1,1,1,1,0,0,1,1,0,1,1,1,0,0,0,0,1,0,1,1,0,0, 1,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,0,1,0,0,1,1,1,0,1,0,1,1,1,0,1,1, 1,1,0,1,1,0,1,0,1,0,1,1,1,0,0,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,0,1,1,1,1,1,0,0,1,1,1,1,0,0,1,1,1, 0,1,1,0,0,1,0,0,0,0,1,0,1,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0,1,1,0,1,1,1,1,1,0,1,0,1,0,1,1,1, 0,0,0,1,0,0,1,0,1,1,1,1,0,1,1,1,0,1,1,0,0,1,1,1,1,0,1,1,1,1,0,1,0,0,0,1,1,0,1,1,0,0,1,0,1,1,1,1, 1,1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,0,1,0,1,1,1,0,1,0,1,0,1,1,1,0,1,0,1,1,1,0,1,0,0,1,1,1,1,0,1,0,1, 0,1,0,1,0,0,0,0,1,1,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,0,1,1,	96.20

	1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,0,1,0,1,1,1,0,1,1,1,1,0,1,1,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,0,1,1,1,1,0,1, 1,1,0,0,1,1,0,1,0,0,1,0,0,0,1,1,0,1,1,1,1,0,1,0,1,0,1,1,1,1,1,1,0,0,1,1,1,0,1,1,1,0,1,0,0,1,0,0, 0,1,1,0,1,1,1,1,0,0,1,1,1,0,1,0,1,1,0,0,1,1,0,0,1,0,1,1,1,1,0,1,1,0,0,0,0,1,0,0,0,1,1,0,0,1,1,1, 1,1,1,1,1,1,0,1,1,1,0,0,0,1,1,1,1,0,0,1,1,1,1,0,0,0,1,1,0,1,1,0,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0, 0,0,0,1,0,1,0,1,0,1,1,1,1,1,1,0,1,0,0,0,1,1,1,1,0,1,0,1,0,1,1,0,1,0,0,1,0,1,1,1,0,1,0,0,0,1,0,1, 0,1,1,1,0,1,1,0,1,1,1,0,1,0,1,1,1,0,1,1,1,0,0,1,1]	
	[0,0,1,0,0,0,0,0,1,0,1,1,1,1,0,1,1,1,1,0,1,1,1,0,1,0,1,1,1,1,0,0,1,1,1,0,0,1,1,1,1,1,1,1,1,1,1, 0,0,0,0,1,1,1,0,1,0,0,1,0,0,0,1,1,0,1,1,0,1,1,1,0,0,1,0,0,1,1,0,1,0,1,0,1,0,0,1,0,0,1,1,0,1,0,0, 0,1,0,1,1,0,1,0,0,1,0,0,1,1,0,0,0,0,0,1,0,0,0,0,1,0,1,1,0,1,0,1,1,0,0,1,0,0,0,1,0,0,1,0,1,1,0,0, 1,0,0,1,0,1,0,0,0,0,1,0,1,0,1,1,0,0,0,1,0,1,1,1,0,0,0,1,0,1,0,1,1,1,0,1,1,0,1,1,1,1,0,0,1,0,0,0, 1,1,1,1,0,0,1,1,1,1,0,0,1,1,1,1,0,1,0,0,0,0,1,1,0,1,0,1,1,0,0,1,0,0,0,0,0,1,1,1,1,1,0,1,0,1,1,0, 0,1,0,0,0,0,1,0,0,1,0,1,0,1,0,1,0,1,0,0,0,1,1,0,0,0,1,0,1,1,1,1,1,1,1,1,1,0,1,1,1,0,1,1,1,0,1,0, 1,1,1,1,1,0,0,0,0,0,1,1,0,1,1,0,0,1,1,0,0,1,0,1,0,0,1,1,0,1,1,0,0,0,1,0,1,1,0,0,0,1,1,1,0,1,0,1, 1,1,1,0,1,1,0,0,0,0,1,0,0,1,0,0,0,1,1,1,0,0,0,0,0,0,1,1,1,0,1,0,0,0,1,1,0,1,0,1,0,1,0,0,0,0,1, 0,1,0,1,1,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,1,1,0,1,0,1,1,0,0,1,1,1,1,1,1,1,1,1,0,0,1,0,1,0,0,0,1, 0,0,1,0,1,1,1,1,1,0,1,0,0,0,0,1,0,0,0,1,1,0,1,0,1,0,0,0,0,0,0,0,1,1,0,1,1,0,0,0,1,1,1,0,0,0,1,1, 1,0,0,1,0,1,0,1,1,1,1,0,0,0,0,0,0,0,1,1,0,0,1,0,1,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1,1,0,1,0,1,1, 1,1,0,1,1,1,1,0,1,0,1,1,1,1,1,1,0,1,1,1,0,0,0,0,1,0,1,1,1,1,1,1,1,1,0,0,0,0,1,1,1,0,0,0,0,1,1,0, 0,1,1,0,1,1,1,0,0,1,1,1,1,0,0,1,1,1,1,1,1,0,0,1,0,0,0,0,1,1,0,0,1,1,1,1,0,0,1,1,0,1,1,0,1,0,0,1, 1,1,0,1,1,0,1,1,1,1,1,0,0,1,0,0,1,1,1,0,1,1,1,0,0]	96

Dataset	**Selected Features by WFLNS on Decision Tree** (five independent runs)	**Accuracy (DT)**
Heart	[1,1,0,1,0,0,1,0,0,0,0,1,1]	81.48
	[0,0,1,0,0,0,0,0,0,0,0,1,1]	85.92
	[0,0,1,0,0,0,0,0,0,0,0,1,1]	81.11
	[0,0,1,0,0,0,0,0,0,0,0,1,1]	85.92
	[0,0,1,0,0,0,0,0,0,0,0,1,1]	81.48
Cleveland	[0,0,0,0,0,0,0,0,1,0,0,1,1]	58.21
	[0,0,0,0,0,0,0,0,1,0,0,0,1]	57.24
	[1,1,0,1,0,0,0,0,1,0,1,1,0]	56.86
	[0,0,1,0,0,0,0,0,1,0,0,0,0]	57.89
	[0,0,1,0,0,0,1,0,1,0,0,1,0]	57.90
Ionosphere	[0,1,1,0,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0,0,0,0,0,1,1,1,0,0,0]	90.25
	[0,0,0,0,1,1,0,0,1,1,0,0,0,1,1,1,0,1,1,0,0,0,1,0,0,0,0,0,0,0,0,1,0]	89.14
	[0,0,0,0,1,1,0,0,1,1,0,0,0,1,1,1,0,1,1,0,0,0,1,0,0,0,0,0,0,0,0,1,0]	89.42
	[1,1,0,0,0,1,1,1,0,0,1,1,1,1,0,1,0,1,1,0,1,0,0,0,1,0,0,0,1,0,1,0,0,0]	88.03
	[1,1,0,0,0,1,1,1,0,0,1,1,1,1,0,1,0,1,1,0,1,0,0,0,1,0,0,0,1,0,1,0,0,0]	91.65
Water	[0,0,0,0,1,1,1,1,1,0,1,0,0,0,1,1,0,0,0,1,1,1,0,1,1,0,1,0,1,1,1,1,1,0,0,0,0,1]	80.23
	[0,0,0,0,1,1,1,1,1,0,1,0,0,0,1,1,0,0,0,1,1,1,0,1,1,0,1,0,1,1,1,1,1,0,0,0,0,1]	81.25
	[1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1]	81.58
	[1,0,0,0,1,0,1,0,1,0,0,0,1,0,1,1,0,0,0,1,1,1,0,1,0,0,1,0,1,0,0,0,1,0,0,0,0,1]	82.32
	[0,0,0,0,1,1,1,1,1,0,1,0,0,0,1,1,0,0,0,1,1,1,0,1,1,0,1,0,1,1,1,1,1,0,0,0,0,1]	81.25
Waveform	[1,0,0,1,1,0,1,0,0,1,0,0,1,1,1,1,0,0,1,0,0,1,1,1,0,0,0,0,0,1,1,0,1,0,0,0,0,1,0,1]	75.38
	[1,0,0,1,1,0,1,0,1,1,0,0,1,1,1,1,1,0,1,0,1,0,0,0,0,0,0,1,0,1,1,1,0,0,0,0,0,0,0,1]	76.81
	[1,0,0,1,1,0,1,0,1,1,0,0,1,1,1,1,1,0,1,0,1,0,0,0,0,0,0,1,0,1,1,1,0,0,0,0,0,0,0,1]	76.66
	[1,0,0,1,1,0,1,0,1,1,0,0,1,1,1,1,1,0,1,0,1,0,0,0,0,0,1,1,0,1,1,1,1,0,0,0,0,0,0,1]	77.40
	[1,0,0,1,1,0,1,0,1,1,0,0,1,1,1,1,1,0,1,0,1,0,0,0,0,0,0,1,0,1,1,1,0,0,0,0,0,0,0,1]	76.66
Sonar	[0,0,0,0,0,0,0,1,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1, 1,0,0,0,0,0,0,0,0,0,0,0]	74.14

	[0,0,0,0,0,0,0,1,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0]	77.54
	[1,1,0,1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,1,1,1,0,1,1,0,0,1,1,1,0,1,1,1,0,1,1,0,1,1,0,0,1,1,1,0,1,1,0,1,0,1,1,1,0,1,1,0,0,1,1]	76.57
	[1,1,0,1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,1,1,1,0,1,1,0,0,1,1,0,0,1,0,1,0,1,1,0,0,1,0,0,1,1,1,0,1,1,0,1,0,1,1,1,0,1,1,0,0,1,1]	77.04
	[1,1,0,1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,1,1,1,0,1,1,0,0,1,1,0,0,1,1,1,0,1,1,0,1,1,0,0,1,1,1,0,1,1,0,1,0,1,1,1,0,1,1,0,0,1,1]	76.57
Ozone	[1,1,1,1,0,0,1,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,1,1,1,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]	95.39
	[1,1,1,1,1,0,0,1,1,0,1,1,0,0,1,1,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,0,1,1,0,0,1,1,1,1,0,1,1,0,0,1,1,0,1,0,1,1,0,1,1,1,1,0,1,1,0,1,1]	95.78
	[1,1,1,1,1,0,0,1,1,0,1,1,0,0,1,1,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,0,1,1,0,0,1,1,1,1,0,1,1,0,0,1,1,0,1,0,1,1,0,1,1,1,1,0,1,1,0,1,1]	96.21
	[1,1,1,1,1,1,1,0,0,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,1,1,0,1,1,1,1,1,0,1,0,0,0,0,1,1,1,1,1,1,1,1,1,0,1,1,0,1,0,1,0,0,0,0,1,0,0,0,0,0,0,1]	95.90
	[0,1,1,1,1,1,1,0,0,0,0,0,0,1,1,0,0,1,0,0,0,0,0,0,0,1,0,0,1,1,1,1,0,1,0,1,1,1,0,1,0,0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,0,1,1,0,1,1,0,1,0,0,1,0,1,1,1,0]	95.23
Libras	[1,1,1,0,1,0,1,1,1,0,1,0,1,1,1,1,1,0,0,1,0,1,1,1,1,1,1,1,1,0,0,0,1,0,1,1,0,1,1,0,1,0,0,1,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,0,1,1,0,0,1,1,1,1,1,1,1,1,1]	68.05
	[1,1,1,0,1,0,1,1,1,0,1,0,1,1,1,1,1,0,0,1,0,1,1,1,1,1,1,1,1,0,0,0,1,0,1,1,0,1,1,0,1,0,0,1,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,0,1,1,0,0,1,1,1,1,1,1,1,1,1]	68.33
	[0,1,0,1,0,1,0,0,0,1,0,0,0,0,1,1,1,1,0,0,0,1,1,1,1,0,1,1,0,0,0,0,1,1,0,1,1,1,1,1,0,1,0,1,0,1,1,0,1,1,1,1,1,0,1,0,1,1,0,1,0,1,1,1,1,1,1,0,1,0,0,0,0,0,1,0,1,0,0,0,1,1,1,0,1,1,0,1,0,1]	69.72
	[0,1,0,1,0,1,1,0,1,1,1,0,0,0,1,1,1,1,0,1,0,1,1,1,1,0,1,1,0,0,0,0,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,1,0,1,0,0,0,1,1,1,0,1,1,1,1,0,1]	70.27
	[1,1,0,1,1,1,1,0,1,1,1,0,0,0,1,1,1,1,0,1,0,1,1,1,1,0,1,1,0,0,0,0,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,1,0,1,0,0,0,1,1,1,0,1,1,1,1,0,1]	70.55

Arrhythmia	[0,1,0,0,1,0,0,1,1,1,0,0,1,0,1,0,0,1,0,1,1,0,0,0,0,1,1,0,1,0,0,1,0,1,0,0,0,0,1,0,0,0,1,0,0,0,1,0,1,0,1,0,1,0,1,0,0,0,0,1,0,1,1,0,1,1,1,0,0,0,1,0,1,0,1,0,1,1,1,0,1,0,0,0,1,0,1,1,0,1,1,1,0,0,0,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,0,1,1,0,1,1,1,1,1,1,1,0,0,0,0,1,1,0,1,1,0,1,0,1,1,1,1,0,0,1,0,1,0,1,0,0,1,0,0,0,0,0,0,1,0,0,1,0,1,1,1,0,1,1,1,1,1,0,0,1,1,1,1,0,1,0,1,0,0,0,0,0,0,1,1,1,0,1,0,0,1,1,1,1,0,1,1,0,0,0,1,0,1,0,1,1,1,0,1,0,1,1,1,1,1,1,1,1,0,1,1,0,1,0,1,0,0,1,1,0,1,1,1,0,1,1,1,1,0,0,1,1,1,0,1,0,1,1,0,0,1,1,1,1,1,0,1,0,1,0,1,1,1,1,1,0,1,0,0,0,1,1,1]	67.54
	[0,1,0,0,1,0,0,1,1,1,0,0,1,0,1,0,0,1,0,1,1,0,0,0,0,1,1,0,1,0,0,1,0,1,0,0,0,0,1,0,0,0,1,0,0,0,1,0,1,0,1,0,1,0,1,0,0,0,0,1,0,1,1,0,1,1,1,0,0,0,1,0,1,0,1,0,1,1,1,0,1,0,0,0,1,0,1,1,0,1,1,1,0,0,0,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,0,1,1,0,1,1,1,1,1,1,1,0,0,0,0,1,1,0,1,1,0,1,0,1,1,1,1,0,0,1,0,1,0,1,0,0,1,0,0,0,0,0,0,1,0,0,1,0,1,1,1,0,1,1,1,1,1,0,0,1,1,1,1,0,1,0,1,0,0,0,0,0,0,1,1,1,0,1,0,0,1,1,1,1,0,1,1,0,0,0,1,0,1,0,1,1,1,0,1,0,1,1,1,1,1,1,1,1,0,1,1,0,1,0,1,0,0,1,1,0,1,1,1,0,1,1,1,1,0,0,1,1,1,0,1,0,1,1,0,0,1,1,1,1,1,0,1,0,1,0,1,1,1,1,1,0,1,0,0,0,1,1,1]	68.41
	[0,1,0,0,1,0,0,1,1,1,0,1,1,0,1,0,0,1,0,1,1,0,0,0,0,1,1,0,1,0,0,1,0,1,0,0,0,0,1,0,0,0,1,0,0,0,1,0,1,0,1,0,1,0,1,0,0,0,0,1,0,1,1,0,1,1,1,0,0,0,1,0,1,1,1,0,1,1,1,0,1,0,0,0,1,0,1,1,0,1,1,1,0,0,0,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,0,1,1,0,1,1,1,1,1,1,1,0,0,0,0,1,1,0,1,1,0,1,0,1,1,1,1,0,0,1,0,1,0,1,0,0,1,0,0,0,0,0,0,1,0,0,1,0,1,1,1,0,1,1,1,1,1,0,0,1,1,1,1,0,1,0,1,0,0,0,0,0,0,1,1,1,0,1,0,0,1,1,1,1,1,1,1,0,0,0,1,1,1,0,1,1,1,0,1,0,1,1,1,1,1,1,1,1,0,1,1,0,1,0,1,0,0,1,1,0,1,1,1,0,1,1,1,1,0,0,1,1,1,0,1,0,1,1,0,0,1,1,1,1,1,0,1,0,1,1,1,1,1,1,1,0,1,0,0,1,1,1,1]	68.85
	[0,0,0,0,1,0,0,0,1,0,0,0,1,0,1,0,0,0,0,1,1,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,1,1,0,0,1,0,1,0,1,0,0,0,0,0,1,0,1,1,0,0,1,1,0,0,0,1,0,1,1,1,0,0,1,0,1,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,1,1,0,0,1,0,1,0,1,1,0,1,1,0,0,0,0,0,0,1,0,1,0,1,0,1,1,0,0,0,1,0,0,1,0,1,1,0,1,1,1,0,0,0,1,0,1,0,0,1,1,0,0,0,0,0,1,1,1,0,0,1,0,0,0,0,0,0,0,1,1,1,0,0,0,1,0,1,0,1,0,0,0,0,0,0,0,0,1,1,0,0,1,0,0,1,1,1,0,1,1,1,0,1,0,1,1,1,0,0,1,1,0,0,0,0,1,1,0,1,1,0,1,0,0,1,0,0,0,0,0,0,1,0,0,1,1,1,0,1,0,1,0,0,0,1,1,0,0,1,0,1,0,1,0,1,1,0,1,0,0,1,0,1,1,0,0,1,0,0,0,1,1,0,1,1,0,0]	68.41
	[0,0,0,0,1,0,0,0,1,0,0,0,1,0,1,0,0,0,0,1,1,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,1,1,0,0,1,0,1,0,1,0,0,0,0,0,1,0,1,1,0,0,1,1,0,0,0,1,0,1,1,1,0,0,1,0,1,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,1,1,0,0,1,0,1,0,1,1,0,1,1,0,0,0,0,0,0,1,0,1,0,1,0,1,1,0,0,0,1,0,0,1,0,1,1,0,1,1,1,0,0,0,1,0,1,0,0,1,1,0,0,0,0,0,1,1,1,0,0,1,0,0,0,0,0,0,0,1,1,1,0,0,0,1,0,1,0,1,0,0,0,0,0,0,0,0,1,1,0,0,1,0,0,1,1,1,0,1,1,1,0,1,0,1,1,1,0,0,1,1,0,0,0,0,1,1,0,1,1,0,1,0,0,1,0,0,0,0,0,0,1,0,0,1,1,1,0,1,0,1,0,0,0,1,1,0,0,1,0,1,0,1,0,1,1,0,1,0,0,1,0,1,1,0,0,1,0,0,0,1,1,0,1,1,0,0]	68.62

Handwritten	[1,1,1,1,1,0,1,0,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,0,1,1,1,0,0,1,1,1,1,1, 1,1,1,0,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,0,1,1,0,1,0,0,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,0,1,1,1,1,1,1,1,1, 0,0,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0, 1,1,0,1,0,0,0,1,0,1,1,1,1,1,1,1]	77.61
	[1,1,1,0,1,0,1,0,1,1,0,1,1,0,1,1,1,1,1,0,0,1,1,1,1,0,0,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1, 0,1,1,1,1,1,0,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0, 1,1,1,1,1,1,0,0,1,1,1,0,1,1,0,1,0,0,1,1,1,0,0,0,1,1,0,1,1,0,1,0,1,1,0,1,1,1,0,0,1,0,0,1,1,1,1,1, 0,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,0,1,1,0,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,0, 1,1,1,1,0,1,1,1,0,0,1,1,1,1,1,1,1,1,1,1,0,1,1,0,1,1,1,0,0,1,1,1,1,0,1,1,1,1,1,1,1,1,0,1,1,0,0,1, 1,1,0,1,0,0,0,1,0,1,1,1,1,1,1,1]	76.79
	[1,1,1,1,1,0,1,0,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,0,1,1,1,0,0,1,1,1,1,1, 1,1,1,0,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,0,1,1,0,1,0,0,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,0,1,1,1,1,1,1,1,1, 0,0,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0, 1,1,0,1,0,0,0,1,0,1,1,1,1,1,1,1]	77.42
	[1,1,1,1,1,0,1,0,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,0,1,1,1,0,0,1,1,1,1,1, 1,1,1,0,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,0,0,1,1,1,1,0,1,1,1,1,0,1,1,0,1,0,0,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,0,1,1,1,1,1,1,1,1, 0,0,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0, 1,1,0,1,0,0,0,1,0,1,1,1,1,1,1,1]	77.11
	[1,1,1,1,1,0,1,0,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,0,1,1,1,0,0,1,1,1,1,1, 1,1,1,0,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,0,0,1,1,1,1,0,1,1,1,1,0,1,1,0,1,0,0,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,0,1,1,1,1,1,1,1,1, 0,0,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0, 1,1,0,1,0,0,0,1,0,1,1,1,1,1,1,1]	77.29

Secom	[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,1,0, 1,1,0,1,0,1,0,1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,1,1,0,0,1,0,1,1,0,1,1,1,1,0,1,0,0,1,1, 0,1,0,1,1,0,0,0,0,1,0,1,0,0,1,0,1,1,1,0,0,1,0,1,1,1,1,1,1,0,1,0,0,1,1,1,0,1,1,0,1,0,0,0,0,0,1,1, 1,0,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,1,0,1,1,0,0,1,0,0,1,0,1,0,0,0,1,1,1,0,1,0,0,0,0,0,0,1,0,1,1, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,1,1,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0,1,1,0,0,1,1,1,0, 1,0,1,0,0,1,0,1,1,0,1,0,0,1,1,0,1,0,0,1,1,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,1,0,0,1,0, 0,0,1,0,0,1,0,0,1,0,1,1,0,1,1,1,1,0,0,1,1,0,0,0,0,1,0,1,1,0,0,0,1,0,1,0,1,1,1,1,0,0,1,0,1,0,1, 1,0,0,0,0,0,0,1,0,0,1,1,1,1,1,0,0,0,0,1,0,1,0,1,0,1,0,1,1,1,0,1,1,0,1,1,1,1,0,1,0,0,1,1,1,0,0,1, 1,0,0,0,0,1,1,0,1,0,0,0,0,1,1,1,0,0,1,1,0,1,0,1,0,0,0,0,1,1,0,0,0,0,1,0,1,1,0,0,0,0,0,1,0,0,1,1, 1,0,0,1,1,1,1,1,0,0,1,0,1,0,0,0,1,1,0,1,0,1,1,1,0,1,1,0,0,0,0,0,1,0,1,1,1,0,1,1,0,0,0,0,0,0,0,0, 0,1,1,0,1,0,1,0,0,1,0,1,0,1,0,0,1,1,0,1,1,1,0,1,0,0,1,0,0,1,0,0,1,0,1,1,1,0,0,1,1,0,0,0,0,0,0,0, 0,0,1,0,0,0,0,0,0,1,1,0,0,0]	90.37
	[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,1,0, 1,1,0,1,0,1,0,1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,1,1,0,0,1,0,1,1,0,1,1,1,1,0,1,0,0,1,1, 0,1,0,1,1,0,0,0,0,1,0,1,0,0,1,0,1,1,1,0,0,1,0,1,1,1,1,1,1,0,1,0,0,1,1,1,0,1,1,0,1,0,0,0,0,0,1,1, 1,0,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,1,0,1,1,0,0,1,0,0,1,0,1,0,0,0,1,1,1,0,1,0,0,0,0,0,0,1,0,1,1, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,1,1,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0,1,1,0,0,1,1,1,0, 1,0,1,0,0,1,0,1,1,0,1,0,0,1,1,0,1,0,0,1,1,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,1,0,0,1,0, 0,0,1,0,0,1,0,0,1,0,1,1,0,1,1,1,1,0,0,1,1,0,0,0,0,1,0,1,1,0,0,0,1,0,1,0,1,1,1,1,0,0,1,0,1,0,1, 1,0,0,0,0,0,0,1,0,0,1,1,1,1,1,0,0,0,0,1,0,1,0,1,0,1,0,1,1,1,0,1,1,0,1,1,1,1,0,1,0,0,1,1,1,0,0,1, 1,0,0,0,0,1,1,0,1,0,0,0,0,1,1,1,0,0,1,1,0,1,0,1,0,0,0,0,1,1,0,0,0,0,1,0,1,1,0,0,0,0,0,1,0,0,1,1, 1,0,0,1,1,1,1,1,0,0,1,0,1,0,0,0,1,1,0,1,0,1,1,1,0,1,1,0,0,0,0,0,1,0,1,1,1,0,1,1,0,0,0,0,0,0,0,0, 0,1,1,0,1,0,1,0,0,1,0,1,0,1,0,0,1,1,0,1,1,1,0,1,0,0,1,0,0,1,0,0,1,0,1,1,1,0,0,1,1,0,0,0,0,0,0,0, 0,0,1,0,0,0,0,0,0,1,1,0,0,0]	89.41
	[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,	89.54

	1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,0,0,0,0,0,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,0,0,0,0,0,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,1,0,0, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0, 0,0,1,1,1,0,0,0,1,0,0,0,1,1]	
	[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,0,0,0,0,0,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,0,0,0,0,0,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,1,0,0, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0, 0,0,1,1,1,0,0,0,1,0,0,0,1,1]	89.54
	[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,1,0,0,1,0, 1,1,0,1,0,1,0,1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,1,1,0,0,1,0,1,1,0,1,1,1,1,0,1,0,0,1,1, 0,1,0,1,1,0,0,0,0,1,0,1,0,0,1,0,1,1,1,0,0,1,0,1,1,1,1,1,1,0,1,0,0,1,1,1,0,1,1,0,1,0,0,0,0,0,1,1, 1,0,0,1,0,0,0,0,0,1,0,1,0,0,0,0,1,0,1,1,0,0,1,0,0,1,0,1,0,0,0,1,1,1,0,1,0,0,0,0,0,0,0,1,0,1,1, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,1,1,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0,1,1,0,0,1,1,1,0, 1,0,1,0,0,1,0,1,1,0,1,0,0,1,1,0,1,0,0,1,1,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,1,0,0,1,0, 0,0,1,0,0,1,0,0,1,0,1,1,0,1,1,1,1,0,0,1,1,0,0,0,0,0,1,0,1,1,0,0,0,1,0,1,0,1,1,1,1,0,0,1,0,1,0,1,	90.37

	1,0,0,0,0,0,1,0,0,1,1,1,1,1,0,0,0,1,0,1,0,1,0,1,0,1,1,1,0,1,1,0,1,1,1,1,0,1,0,0,1,1,1,0,0,1, 1,0,0,0,0,1,1,0,1,0,0,0,0,1,1,1,0,0,1,1,0,1,0,1,0,0,0,1,1,0,0,0,0,1,0,1,1,0,0,0,0,0,1,0,0,1,1, 1,0,0,1,1,1,1,1,0,0,1,0,1,0,0,0,1,1,0,1,0,1,1,1,0,1,1,0,0,0,0,1,0,1,1,1,0,1,1,0,0,0,0,0,0,0, 0,1,1,0,1,0,1,0,0,1,0,1,0,1,0,0,1,1,0,1,1,1,0,1,0,0,1,0,0,1,0,0,1,0,1,1,1,0,0,1,1,0,0,0,0,0,0, 0,0,1,0,0,0,0,0,0,1,1,0,0,0]	
Multifeat	[1,1,0,0,1,1,0,0,0,0,0,1,1,0,0,1,1,0,1,1,1,1,1,0,1,0,1,1,0,0,1,0,1,1,1,0,0,1,1,0,1,1,1,1,0,0,1, 1,1,1,1,0,0,1,1,0,0,0,1,0,1,0,0,1,0,0,1,0,0,0,1,1,0,1,0,0,1,1,1,0,1,1,1,1,0,0,0,1,0,1,1,0,0,1, 1,1,1,1,1,1,1,1,0,1,1,1,0,1,1,1,0,1,0,1,1,0,0,1,1,0,0,0,0,1,0,1,1,0,1,1,1,1,0,0,0,1,1,1,1,0,0,1, 1,1,0,1,0,1,1,1,1,1,0,0,1,1,1,1,0,1,1,1,0,1,0,1,0,1,0,1,0,0,0,0,1,0,1,0,1,1,1,1,1,1,0,1,1,1,0,0, 1,1,1,0,1,1,1,0,0,0,1,0,0,1,1,1,1,1,1,1,1,1,1,1,0,1,1,0,0,0,1,0,0,1,0,1,1,1,1,0,0,1,0,1,1,0,1,1, 0,0,0,1,0,1,1,0,1,0,1,1,1,1,0,1,1,1,1,0,1,1,1,0,1,1,1,0,1,0,1,0,1,0,1,0,1,0,1,1,0,0,0,1,1,1,1,0, 1,0,0,0,0,1,1,1,1,1,0,1,0,1,1,0,0,1,0,0,0,1,1,1,1,0,1,1,0,1,1,1,0,0,0,0,0,0,0,1,1,0,1,0,0,1,0, 0,0,1,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,1,1,0,1,1,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,1,1,0, 0,1,1,0,0,0,1,1,0,0,1,1,1,0,0,0,0,1,1,1,0,0,0,0,1,1,1,1,0,1,1,0,0,1,1,1,1,0,0,1,1,1,0,1,1,0,1,0, 1,0,1,1,0,1,1,1,1,0,1,1,0,1,1,0,1,1,1,0,1,1,0,1,0,1,0,1,0,1,0,1,1,1,0,1,1,0,1,0,1,0,1,0,1,0, 1,1,1,1,1,0,0,1,1,0,1,0,0,1,0,1,1,1,0,1,0,1,1,1,1,1,1,0,1,1,0,1,0,0,0,1,1,0,1,0,1,1,0,1,1,1,0,0, 1,0,1,1,0,1,0,0,0,1,1,1,1,0,0,1,1,1,1,0,1,1,1,0,1,1,1,1,0,0,1,1,0,1,1,0,0,0,1,1,1,1,0,0,0,0,1,1, 0,1,1,1,1,1,1,0,0,0,1,0,0,0,0,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0,1,0,1,1,1,1,1,0,1,0,1, 0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0,0,0,0,1,0,1,0,0,1]	93.15
	[0,0,1,1,0,1,0,0,1,1,1,0,1,1,0,0,1,1,1,1,1,1,0,1,1,1,1,1,1,0,1,0,0,0,1,0,1,0,0,0,0,1,0,0,1,0,0,1, 1,1,1,1,1,1,0,0,0,1,0,1,0,1,1,0,1,1,0,0,0,1,0,0,1,1,0,0,1,1,0,1,1,1,0,0,1,1,1,1,0,1,1,0,1,1,0,1, 1,1,1,0,1,1,1,0,1,1,1,0,0,0,1,1,1,1,1,1,1,1,0,1,1,0,1,0,1,1,1,0,1,1,1,0,1,1,1,0,0,1,0,1,0,0,1,1, 1,1,1,0,0,1,1,1,1,1,1,0,1,1,1,0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0,0,1,1,1,0,1,1,1,1,1,1,1,1,1,0,1,1, 1,0,0,1,0,0,0,0,1,0,1,1,1,1,1,0,1,1,1,0,1,1,1,1,0,1,1,0,1,0,1,0,1,1,1,1,0,1,1,1,0,1,0,1,0,0,0,1, 1,0,1,1,0,1,1,0,1,0,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,0,1,1,1,1,0,0,0,1,0,0,1,1,1,1,1,1, 0,1,0,1,1,1,1,1,1,1,0,1,1,1,1,0,0,1,0,0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0,0,1,0,0,0,1,1,0,1,0,0,1,1, 1,0,1,1,1,1,1,0,0,1,0,0,1,0,1,0,0,1,0,0,1,0,1,0,0,0,1,1,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,1,1,1,1, 1,1,1,1,0,0,1,1,0,1,1,1,1,0,0,0,0,1,1,1,1,0,1,0,1,1,0,1,0,1,1,1,0,1,1,1,1,0,1,1,1,0,0,1,1,0,1,0, 1,1,0,1,0,1,1,0,1,0,1,1,1,1,1,0,0,1,1,1,1,0,1,1,0,1,0,0,1,1,0,1,0,1,1,1,1,0,1,0,0,0,0,0,1,0,1,1,	92.60

	0,1,1,0,0,1,1,1,0,1,0,1,1,1,1,0,1,1,1,1,0,1,1,0,0,0,1,0,1,1,0,1,1,0,1,0,1,1,0,1,0,1,0,1,0,1,1,1, 1,0,1,1,0,1,0,1,1,1,1,0,1,0,0,1,1,1,1,0,1,1,1,0,1,1,1,1,0,0,1,1,1,0,1,0,0,0,1,1,1,1,0,1,0,1,1,1, 1,0,0,1,1,1,1,1,0,1,1,0,0,0,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,0, 0,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,1,1,0,1]	
	[0,1,0,1,1,0,0,0,1,1,1,0,1,1,0,1,0,1,1,0,1,0,1,1,0,1,1,0,0,0,1,1,0,1,1,1,0,1,1,1,0,0,1,1,1,0,1,0, 0,1,1,1,0,0,1,0,0,0,1,0,0,1,1,0,0,0,1,1,1,1,1,0,0,1,0,1,0,1,1,1,1,0,1,1,0,0,0,1,0,1,1,1,0,1,0,0, 0,0,1,1,0,1,0,0,0,0,1,0,1,1,1,0,0,1,0,0,1,0,1,0,0,1,0,0,0,1,1,1,0,0,1,0,1,1,1,0,0,1,0,1,0,1,1, 1,1,1,1,0,0,1,1,0,0,1,0,0,0,0,1,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0,1,0,0,1,0,0,0,1,1,0,1,1,0,1,1,0,0, 0,1,1,1,1,1,0,1,0,0,1,0,1,1,1,1,1,0,0,1,1,0,0,1,1,1,0,0,0,1,1,0,1,0,1,0,1,1,0,1,0,1,0,0,1,0,1,0, 0,1,0,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,1,0,1,1,1,0,1,0,1,0,1,1, 0,0,1,0,1,1,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1,0,0,0,0,1,1,0,0,1,1,0,0, 0,1,1,0,0,0,1,1,1,0,0,1,1,1,0,0,1,1,1,0,0,1,1,1,0,1,1,1,1,1,0,1,1,1,0,0,1,1,1,0,1,1,0,1,1,0,1,1, 1,0,0,1,1,1,0,1,1,1,1,1,0,1,1,1,0,0,1,0,1,0,0,1,0,1,1,0,0,1,1,0,0,1,0,1,0,0,1,1,1,0,0,1,1,1,0,0, 1,1,1,0,0,1,1,1,0,0,1,1,0,0,0,0,1,1,0,0,0,1,0,0,0,1,1,0,0,0,0,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,1, 1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,0,1,0,1,1,1,1,1,1,0,0,0,1,0,1,1,1,1, 1,1,0,0,0,1,0,1,1,1,1,0,0,0,0,1,1,1,0,0,0,1,0,0,1]	89.60
	[1,0,0,0,0,0,0,1,0,1,0,1,0,0,0,1,1,1,1,0,0,1,0,1,0,0,1,1,0,1,0,0,1,0,0,1,0,0,1,1,0,1,0,1,0,1,0,0, 0,0,1,1,0,1,0,1,1,1,0,0,0,1,0,1,0,0,0,0,0,0,0,0,1,0,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,1,0,1,0,1,1, 0,0,0,1,1,0,1,1,1,1,1,1,1,0,1,1,0,1,0,0,1,0,0,0,0,1,1,1,1,1,0,1,0,0,1,0,1,0,1,0,0,1,1,0,0,1,0,0, 0,1,0,1,0,1,0,1,0,1,1,0,1,0,1,0,1,0,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,1,1,0,1,1,0,0,1,0,1,1,0,0,1, 1,1,0,1,0,1,1,0,1,1,1,1,0,1,1,1,1,0,1,1,1,0,0,0,1,0,1,0,1,0,1,1,1,0,0,1,0,1,0,1,1,0,0,0,0,1,1,0, 1,0,0,1,1,1,0,0,1,0,0,0,1,1,0,1,1,1,0,0,0,1,1,1,1,0,1,0,0,1,0,0,1,1,0,1,1,1,0,1,0,1,0,1,0,1,1,0, 1,1,1,1,0,0,1,1,0,1,0,1,0,1,0,1,1,1,1,0,1,0,1,0,1,0,0,0,1,1,0,1,0,0,1,1,0,0,1,1,1,0,0,0,0,0,0,0, 0,0,0,1,0,1,0,0,1,0,1,1,1,1,0,1,0,0,1,0,0,1,0,1,0,1,0,0,0,1,1,0,1,0,1,0,0,0,0,1,1,1,1,0,0,1,1,0, 0,0,0,0,1,1,1,0,0,0,0,1,0,1,0,1,1,1,1,1,1,1,1,0,0,1,1,0,0,0,0,1,1,0,1,0,0,0,1,0,1,1,1,1,0,0,0,1, 1,1,1,0,0,0,0,0,1,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,1,0,1,1,0,1,1,1,0,0,0,0,0,0,1,1,1,0,0,0,0,0,1,1, 0,1,1,0,1,0,0,0,0,1,1,1,0,1,1,0,0,1,0,0,0,0,0,0,1,1,0,0,1,1,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,0,1,1, 1,1,0,0,0,1,1,1,0,1,1,1,0,1,0,1,1,0,0,0,0,1,1,0,0,1,0,1,0,0,1,1,0,1,1,1,0,0,0,0,1,1,1,1,0,0,0,1,	92.85

	0,0,0,1,1,0,0,1,1,1,1,1,0,0,0,1,0,1,1,0,0,0,0,1,1,1,0,0,0,0,0,1,0,1,1,1,1,1,0,0,1,0,0,1,1, 0,1,1,1,1,1,1,1,0,1,0,0,1,1,0,0,0,1,0,0,1,1,0,1,1]	
	[1,0,0,1,1,0,0,1,0,0,0,0,1,1,0,1,0,1,0,0,0,0,1,0,0,1,0,0,0,0,1,1,0,1,1,1,0,1,0,1,0,1,0,0,1,0,1,1, 1,1,1,0,0,0,0,0,0,1,0,0,1,1,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,1,1,0,0,0,0,0,0,1,0,1,1,0,0,1,0,1, 0,1,0,1,0,1,1,1,0,1,0,1,0,1,0,0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,1,0,1,0,0,0,1,1,0,1,0,0,1,1,1,0,0,1, 0,1,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,1,1,0,1,1,1,0,0,0,1,1,0,1,0,0,0,1,1,1,1,0,1,1,0,0,0, 1,0,1,1,0,0,1,0,0,0,0,1,1,1,1,1,1,0,0,1,0,0,0,1,1,0,0,0,0,1,1,0,0,1,1,1,1,0,1,0,0,1,0,0,1,0,0,0, 0,1,0,0,0,1,0,0,0,0,0,1,0,0,0,1,1,0,1,0,0,0,1,0,0,1,1,0,1,0,1,0,0,0,0,0,1,0,0,0,1,0,1,0,1,0,0,1, 1,0,0,0,1,1,1,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,1,0,0,1,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,1,0,0,1,1,1,0,0,0,0,0,1,0,0,0,0,0,1,1,1,0,0,0,1,0,0,0,1,0,0,0,0,1,0,1,0,0,0,1,0,0, 0,1,0,0,0,0,1,1,1,0,1,1,1,1,1,0,1,1,0,0,0,1,0,1,0,1,0,0,1,1,0,1,0,0,0,0,1,1,0,0,1,0,1,0,1,0,0,0, 0,0,0,0,1,0,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,1,1,0,0,0,0,0,1,0,1,0,1,1,0,1,0,1,0,0, 0,1,0,1,0,1,1,1,0,0,0,0,0,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,1,1,0,1,0,0,0,0,0,0,1,0,0,0,0,1,1, 1,0,0,0,1,0,1,0,0,0,1,0,0,0,0,1,0,1,0,1,0,0,0,0,1,1,1,0,1,0,0,1,0,0,1,1,1,1,0,0,1,0,1,1,1,0,1,0, 1,1,0,1,0,0,0,0,1,0,1,0,0,1,0,0,1,1,0,0,0,1,0,0,1]	92.95