Classification of multiclass datasets using genetic programming

Conference Paper · September 2019

DOI: 10.1145/3338840.3355656

3 authors, including:

Mahsa Shokri Varniab

5 PUBLICATIONS **6** CITATIONS

SEE PROFILE

Vahid Khalilzad-Sharghi

Georgia Institute of Technology

28 PUBLICATIONS **284** CITATIONS

SEE PROFILE

Classification of Multiclass Datasets Using Genetic Programming

Mahsa Shokri Varniab
Kennesaw State University
Marietta, Georgia
USA
mshokriv@students.kennesaw.edu

Chih-Cheng Hung
Kennesaw State University
USA and
Anyang Normal University, China
chung1@kennesaw.edu

Vahid Khalilzad Sharghi
Kennesaw State University
Marietta, Georgia
USA
vkhalilz@kennesaw.edu

ABSTRACT

This paper[1] proposes an approach which uses optimized genetic programming (GP) with a new fitness function for multiclass dataset classification. In place of defining static thresholds as boundaries to differentiate between multiple classes, our work presents a method of classification where a GP system learns the relationships among experiential data and models them mathematically during the evolutionary process. We propose an optimized GP classifier based on a combination of pruning subtrees and a new fitness function. An orthogonal least squares algorithm is also applied in the training phase to create a robust GP classifier. Our approach has been assessed on four multiclass datasets and compared against three existing methods. The analyzed results illustrate that the developed classifier produces a productive and rapid method for classification tasks that outperforms the previous methods for more challenging multiclass classification problems.

CCS CONCEPTS

• **Computing methodologies → Machine Learning**;

KEYWORDS

Classification, Genetic Programming, Machine Learning

ACM Reference format:
M. Shokri Varniab, CC. Hung, V. Khalilzad Sharghi. 2019. Classification of Multiclass Datasets Using Genetic Programming. In *Proceedings of ACM RACS, Chongqing, China, September 24-27, 2019 (RACS'19)*, 7 pages.
https://doi.org/10.1145/3338840.3355656

1 INTRODUCTION

Classification is known as one of the effective data modeling and machine learning techniques [1]. An extensive range of problems in different domains can be solved by classification algorithms. For example, disease diagnosis [2], pattern recognition [3], document categorization [4], credit scoring [5], bankruptcy prediction [6], and software quality assessment [6], to name a few. A classification method uses a training set, including properly labeled data instances and a search algorithm, to induce a classifier from the training set. To determine the excellence of the resulting classifier, a testing set, including a set of properly labeled data instances, is used. Different kinds of models such as decision trees [7] and random forest [8] have been used by researchers to represent classifiers.

In machine learning, two major methods of learning are supervised and unsupervised learning. Methods used in classification and regression are supervised learning. In supervised learning, a set of training dataset with well-labeled classes is used which indicate the correct answers. Hence, we call this type of learning with a teacher. To perform the learning phase, a training dataset including input features and output labels is provided to conduct the learning process. The output in classification is discontinuous while in regression the output is continuous. On the other hand, in unsupervised learning the types of the variables of the dataset are similar. Therefore, we do not have a set of data with a recognized output and there is no teacher for the training. Unsupervised learning leads to discover the inherent configuration, relations, or patterns existing in data. Clustering and association discovery are examples of unsupervised learning tasks [9]. Clustering task categorizes data into distinctive groups, singles out sets of data that are different from each other, and finds which groups' members are similar to one another. Association discovery is the identification of data values that occur together in a given event or record frequently. Association discovery rules are related to occurrence counts of the number of times items take place alone and in combination in the dataset [10].

One of the widely used unsupervised learning techniques is K-means algorithm, which helps to divide n observations into k clusters; however, the weakness of the K-means algorithm is its need for knowing about the number of groups or clusters [11]. This is a big challenge for the data mining tasks because in practice it is difficult to guess the number of clusters properly. Therefore, there is a need to develop an algorithm based on a type of evolutionary algorithm (EA), a subcategory of machine learning, which can be utilized to determine answers to problems that are hard to solve without the help of an intelligent

RACS '19, September 24–27, 2019, Chongqing, China

ACM ISBN 978-1-4503-6843-8/19/09…$15.00
https://doi.org/10.1145/3338840.3355656

machine. EA enables a machine to generate solutions, free of human prejudices or biases, which are equivalent to, and often stronger than a solution developed by human being [12]. GP is a comparatively new and rapidly growing technique to autonomous programming [13]. In GP, problems can be solved in different formats; however, solutions are usually defined as computer programs. The evolution of a population of programs towards an outstanding solution to a problem are carried out based on Darwinian principles of natural selection and recombination. GP is a novel method to tackle a broad range of problems due to the fact that it utilizes the flexibility and fluency of computer program representation as well as the strong proficiencies of evolutionary search. GP applications have showed a trend of success in recent years [14-18].

The focus of this study is mainly to develop a technique based on genetic programing (GP) for classifying datasets precisely without the previous knowledge of numbers of clusters. The developed technique uses a pruning algorithm to promote the accuracy and speed of classification. The GP steps for classification, including the major components of the approach such as terminal set, fitness function, training set and test set are discussed.

2 RELATED WORK

GP has been extensively used to tackle classification problems due to its ability to determine primary data associations. Liu and Xu described GP as the reliable solution to detect and score top-ranked genes as the feature of the experimental data for classification purposes [19-22]. In previous studies, researchers applied GP-based techniques to analyze two-class microarray datasets. The traditional GP system involves evolving tree-based individuals. A tree can generate a binary solution for a classification; therefore, GP is a right method for classifying two-class microarray datasets. Later, this technique was improved to classify multi-class microarray datasets. Liu and Xu showed that multiple-class datasets can be treated as multiple two-class data instances, and a set of sub-group classifiers were utilized to tackle associated two-class data instances. By combining these groups, an individual is generated leading to solve a multiclass problem without a need for a new algorithm. However, this technique can be time consuming and was not tested on a wide range of challenging datasets to be completely verified. GP is also used in other applications such as feature construction [22].

Tahmasebei et al. have used a GP model to classify high activity regions in the Limbic system of the fMRI data. The high dimensionality of fMRI data makes the classification task challenging. In their GP model, a crossover operator was used to select and replace the winner of the tournament with a stochastic subtree. Additionally, their algorithm used mutation to keep the diversity of subtrees. The authors concluded that accuracy of their algorithm is better than typical machine learning algorithms due to the power of the GP method [23]. Despite the authors' preliminary success, this method was designed for a two-class dataset while GP is previously shown to be much more capable for multi-class problems.

In 2015, Al-Sahaf et al. employed GP for multiclass texture classification. In their method, a combination of raw pixel values as inputs and simple mathematical operators was used. The programs generated are used for initialization of a feature vector that is then grown into a nearest neighbor classifier to predict class labels. The performance of their proposed method was evaluated using multiclass datasets. Then, the results were compared with the performances of two GP-based and nine non-GP methods. The authors reported high accuracies for their work. However, their algorithm was not tested on instances with rotation or with different dimensions [24].

3 CLASSIFICATION VIA GP

3.1 Genetic Programming (GP)

GP is an evolutionary algorithm that utilizes concepts learned from biological evolution and finds answers to problems human beings may not know how to solve directly. Each program in GP algorithm is expressed as a chromosome in a population, and each chromosome contests for resources and existence, analogous to natural species contending for resources such as nutrition and dwelling. In GP algorithm, only the most acceptable or near acceptable individuals stay, and they generate newborns in the hope that these newborns can survive [25]. The tree illustration of an example computer program is shown in Fig. 1. Five preliminary steps are taken by an analyst to link the human-level description of the problem to the GP algorithm. These well-defined steps are shown in Fig. 2. The result of the GP algorithm is the best computer program that appears in the process of generations.

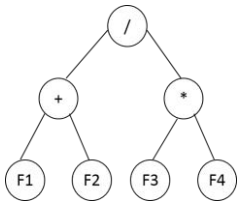

Figure 1: An example computer program for the numerical expression (F1+F2) /(F3*F4).

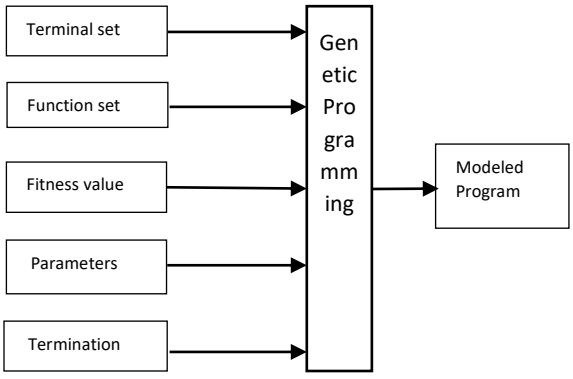

Figure 2: Five major steps in a GP algorithm.

The following steps describe the complete process of the GP system:

i. First, a population is initialized.
ii. The following steps is repeated until an end condition is fulfilled:
    a. Individual programs are evaluated in the present population and a fitness is calculated for them.
    b. The successive tasks are performed in a loop until the next population is completely produced:
        – Select programs and run crossover and mutation operators on them in the current generation.
        – Place the product of the crossover and mutation operators into the new generation.
iii. The best chromosome of the population is provided as the result of the GP system.

GP provides solutions using programs or functions displayed as a tree consisting of primitive functions (internal nodes) and terminals (leaf nodes). The terminals include independent variables and constants, which are the inputs to the problem. GP uses a fitness value which is the basic measure for associating the human-level description of the designer's goals to the GP algorithm and determines a desired goal. Fitness value is used to compare one individual to another and determine how good an individual is [26; 27].

A selection mechanism is employed in GP to select an appropriate evolved program which will be utilized for crossover and mutation operators. The selected programs are employed to create new individuals for the following generation in the period of the evolutionary steps. In this study we used roulette wheel selection (fitness proportionate selection) which is the most commonly used selection method. The roulette method works similar to a simple roulette which randomly rotates and stops at a point. Every single individual possesses a sector of the roulette that links to its foreseen number of offspring.

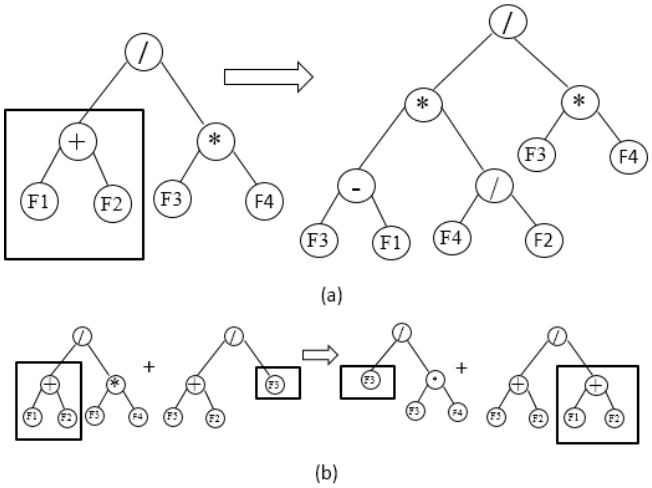

Figure 3: Operations of genetic operators in GP. (a) Mutation; (b) Crossover.

Diversification in the form of mutation and crossover are used for GP systems. Mutation analogous to biological mutation (Fig. 3.a) is utilized to keep genetic diversity in the population. It can also adjust an evolved program by choosing right constants. Furthermore, mutation prevents the population of individuals from getting very similar to each other and therefore creating local minima. Mutation is commonly performed in the form of swap, insert, delete, alter, point, uniform, non-uniform, etc. On the other hand, crossover, which is similar to sexual reproduction, happens between two parents as shown in Fig. 3.b. Crossover recombines the selected parents to generate one, two or more children. Crossover is performed in the form of one-point, two-point, n-point, uniform, and cut-and-splice.

Different control parameters are used for running the GP system. For example, how big the population is, what the probabilities of crossover and mutation are, and how complex the generated programs are. Among them, the population size is the most significant control parameter and needs to be chosen in a way that generates a considerable number of generations within acceptable processing time and complexity.

Termination criteria need to be defined to terminate the GP process when the result is satisfactory. Specific value of fitness function and how many generations the algorithm can proceed are examples of termination criteria. During the GP process, if the value of the fitness does not improve for a specific number of generations, the GP algorithm will stop the process and will pick the individual with the best associated fitness value as the result. Fig. 4 illustrates the basic cycle of GP algorithms.

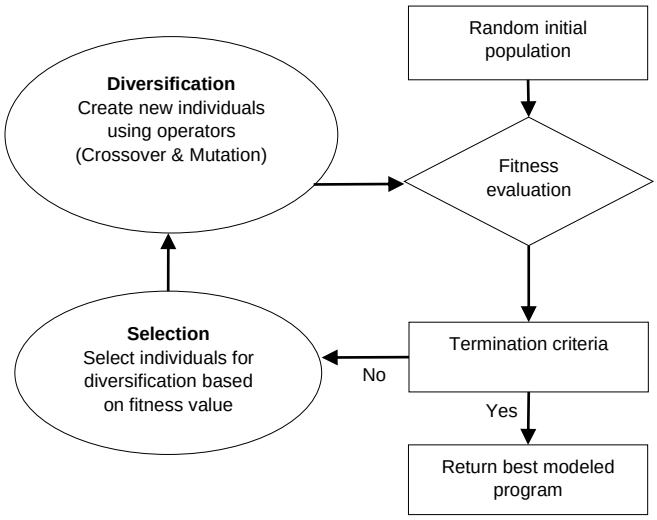

Figure 4: A basic cycle of GP algorithms.

3.2 Classification

In data mining and machine learning, classification is a common method of creating a predictive model for experiential data. The concept of classification is involved with creating a model that partitions data into different classes. The modeled data is created by determining a set of data as training part by which the classes are labeled in order to teach the algorithm. Then, the model is applied on a different dataset called test set in order to predict the class of each member of the dataset using the model learned from training part [15]. In the most cases, the problem uses

supervised training in which a portion of dataset labeled with the type of the class it belongs to is provided to the system.

3.3 GP Classifier

In the past few years, researchers have presented a series of computational techniques for designing new classifiers, such as linear classifiers, quadratic classifiers, k-nearest neighbor, decision trees etc. GP has also been employed because it can discover underlying data relationships [28; 29]. A GP classifier, which uses a set of arithmetic and mathematical operators as well as conditional/logic operators, provides a mathematical equation as the solution to a classification problem. The individual structure for GP classifier is shown in Fig. 5.

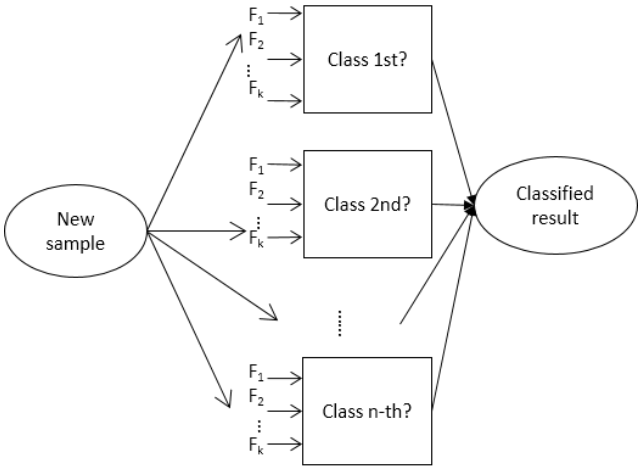

Figure 5: The individual structure for GP classifier.

A fitness function in GP algorithm displays the good quality of a possible solution which depends on the selection probability of the individual. Therefore, we designed a fitness function to guide the GP system to evolve towards a high performing classifier. The new fitness value designed for the n-th individual is shown in equation 1:

$$fitness = \frac{a.TCN}{N + b.FCN} \quad (1)$$

Where *TCN* is true classification number, *FCN* is false classification number, and *N* is the number of instances in the training set. The factors *a* and *b* allow the fitness measure to be adjusted to affect the individuals' sensitivity or specificity. In this design, an individual could be evaluated using the fitness function to measure high value of fitness. Then, if the value of fitness is high, it will be chosen. Also, if more than one individual has the same fitness value, the individual with fewer features will be chosen first.

In this study, an improved GP algorithm which uses a pruning mechanism is utilized to perform the classification with a higher speed and accuracy. In the process of the GP operation, the algorithm produces multiple possible tree-based solutions which are the individual parts of the population. These trees are composed of subtrees with good or bad effects on the accuracy of the model. To improve the GP system, the tree structure is disintegrated to subtrees and the errors of these subtrees are measured. Then the terms with the least importance are removed. This tree pruning step is performed before the calculation of fitness value of the tree as illustrated in Fig. 6. The main purpose of the pruning approach is to simplify the trees and still maintain the accuracies as close as possible to their original trees. Orthogonal least squares (OLS) algorithm is utilized to monitor the decomposition of the trees in order to keep the original structure of trees as much as possible [30]. First, errors of the branches of the tree are calculated and the subtrees with errors less than a threshold are eliminated with respect to OLS algorithm. By using this technique, it is not necessary to rearrange the structure of the tree after pruning. Fitness is calculated in the next step and if it is in the defined range, the associated individual is selected as the model.

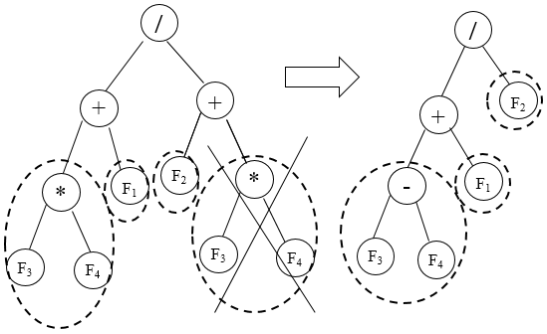

Figure 6: Tree pruning step is performed before the evaluation of fitness value.

4 EXPERIMENTAL RESULTS AND ANALYSIS

4.1 Experiment Setup

In the current study, GP is utilized to classify four types of datasets including Iris, Wine, Glass, and Pima datasets listed in Table 1. The results are compared with other algorithms including Decision Tree (DT), Random Forest (RF), and Random Forest with Self Organizing Map (RF-SOM) [31; 32]. Our GP classifier works in two steps, including training part and testing part. In this study, 70 percent of data is randomly selected for training purpose and the rest is used for testing. For each dataset, 10 experiments with randomly-selected training and testing data are implemented to investigate the accuracy of the genetic programming algorithm. The GP classifier developed in this work provides possible solutions in terms of computer programs consisting of terminal and function parts evolved recursively. The function set used in our GP algorithm contains the primary arithmetic operations (+, -, ×, /) and the terminal set contains the features of each dataset including F_1, F_2, …, and F_k.

Table 1: Four Datasets

Datasets Name	No. Class	No. Features	Dataset Size	No. Each Class
Iris	3	4	150	50+50+50
Wine	3	13	178	59+71+48
Glass	6	9	214	70+76+17+13+9+29
Pima	2	8	768	500+268

A custom-designed fitness function was used to select the best program in the training phase. Then, the best program created during training is applied to classify the test dataset to analyze the accuracy of the GP. Furthermore, the pruning mechanism is applied in the training phase to remove insignificant terms of a generated program, which leads to increase the speed of the GP algorithm and reduce the complexity of programs. Key parameters used in GP developed in this work are shown in Table 2.

Table 2: Parameters utilized in GP

Parameter	Value
Population Size	100
Selection Method	Roulette-wheel
Mutation Operator	Point Mutation
Crossover Operator	One-point Crossover
Proportion of Crossover	70
Proportion of Mutation	30

4.2 Experimental Results

To evaluate the functionality of our proposed GP classifier, we experimented with four datasets including Iris, Wine, Glass, and Pima. MATLAB (MathWorks, Natick, MA) software is used to implement our algorithm, as it is one of the best platforms for numerical and symbolic computing as well as simulation and model-based design. To analyze the classification performance of the model generated at each stage with a higher precision, the accuracy of each dataset is calculated in 10 experiments. Table 3 shows the analysis on accuracies of classification results with our GP system where the columns "Max", "Min" and "Mean" represent the maximum, minimum and average of the overall accuracies of 10 experiments for each dataset.

Table 3: Accuracy resulted by improved GP algorithm

Dataset	Max Accuracy	Min Accuracy	Mean Accuracy
Iris	100	95.55	98.44±1.50
Wine	98.11	94.33	97.54±1.27
Glass	98.43	89.06	93.27±3.21
Pima	83.47	75.65	80.34±2.97

The classification results performed by our developed GP classifier for Iris, Wine, Glass, and Pima datasets are depicted in Figs. 7-10 respectively.

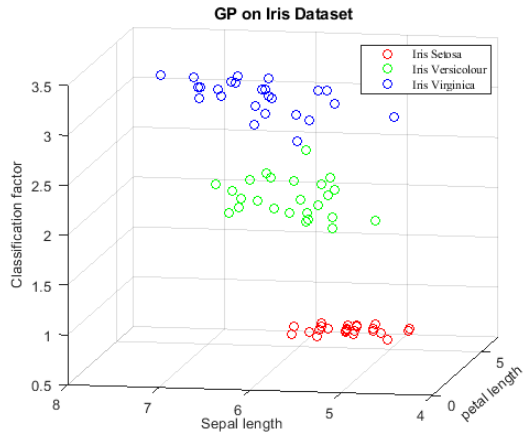

Figure 7: Scatter plot of Iris dataset classification using GP classifier.

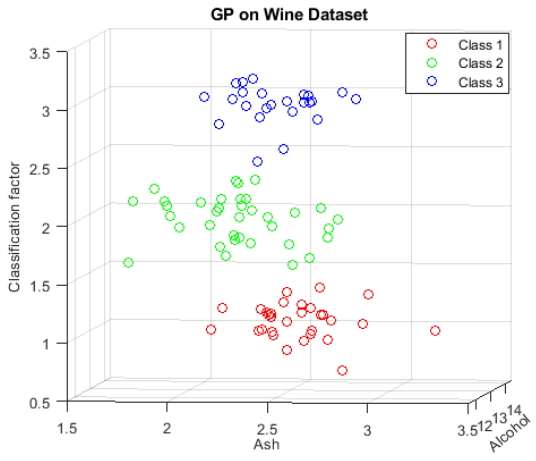

Figure 8: Scatter plot of Wine dataset classification using GP Classifier.

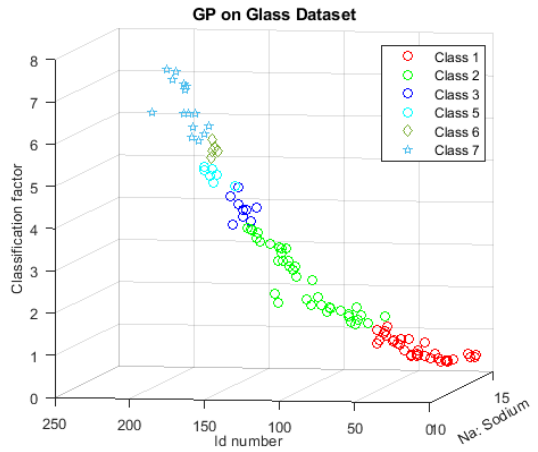

Figure 9: Scatter plot of Glass dataset classification using GP classifier.

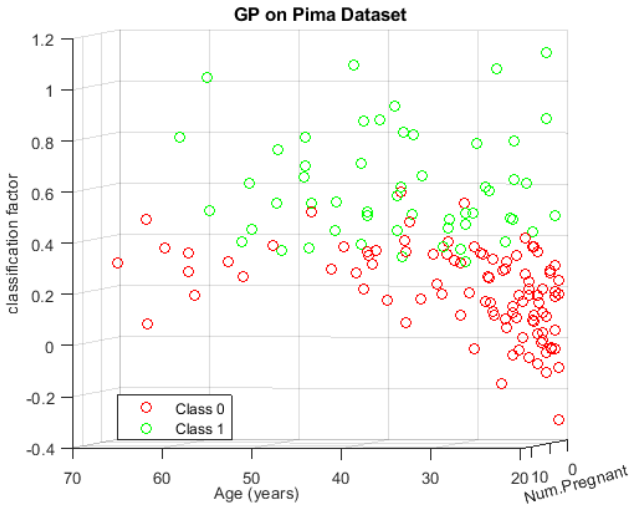

Figure 10: Scatter plot of Pima dataset classification using GP classifier.

The accuracy of the GP classifier developed in this research was compared with those of Decision Tree, Random Forest and Random Forest with Self Organizing Map methods [31; 32] and the results are illustrated in Fig. 11. To evaluate the running speed efficiency, tests are implemented in MATLAB 2019a with a Windows 7 Professional operating system, whose hardware is of an Intel Core i5-2520M 2.50 GHz CPU, 8 GB RAM. The average execution time of the modified GP algorithm for classification of our datasets was approximately 120 seconds. We observed that the average execution time without using pruning with OLS algorithm increased to nearly 270 seconds.

5 CONCLUSION AND FUTURE WORK

In this paper, we propose a novel genetic programming algorithm to classify multiclass datasets. The proposed algorithm uses a pruning mechanism and a new fitness function to solve the classification problem for multiclass datasets. The pruning technique passes orthogonal least squares (OLS) check in order to maintain the original tree-based structure to the extent that it is possible. This is necessary because the tree structure is an essential element of GP system. Our developed GP classifier was tested on Iris, Wine, Glass, and Pima datasets. The results of the four classification problems demonstrate that this method performed very well even when applied on datasets with very small sample sizes. This approach is compared with DT, RF, and RF-SOM. In conclusion, the results indicate that the new GP classifier outperforms the alternative approaches when dealing with datasets with increasing difficulty. In future studies, we will consider whether this approach can improve the performance of object detection on image datasets, particularly for tumor detection on brain magnetic resonance imaging (MRI) images.

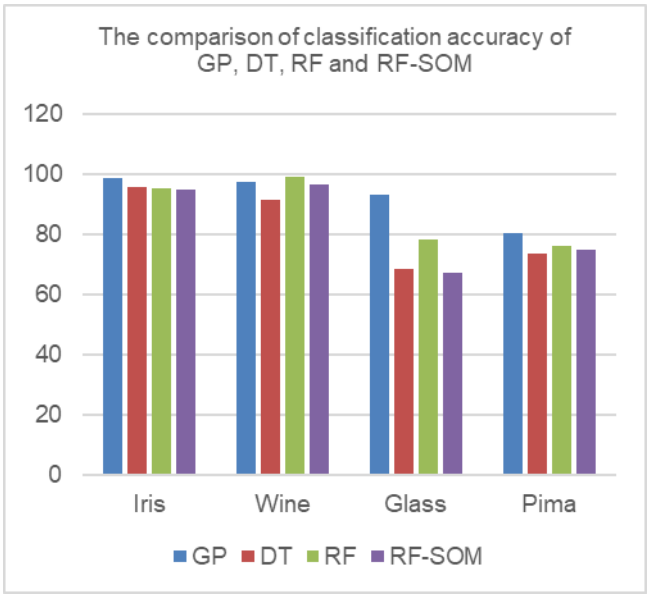

Figure 11: The comparison of classification accuracy for each dataset using GP, DT, RF, and RF-SOM.

ACKNOWLEDGEMENT

Our thanks go to the Graduate College at Kennesaw State University in Kennesaw, Georgia for supporting this research.

REFERENCES

[1] J. Han, M. Kamber, and J. Pei, 2011. Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers Inc.
[2] V. Khalilzad-Sharghi, A. Talebpour, A. Kamali-Asl, and N. Hendijani, 2008. Automatic Assessment of Cardiac Artery Disease by Using DCAD Module. In 2008 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, 243-246. DOI= http://dx.doi.org/10.1109/SYNASC.2008.57.
[3] P. Sharma and M. Kaur, 2013. Classification in pattern recognition: A review. International Journal of Advanced Research in Computer Science and Software Engineering 3, 4.
[4] L. Cai and T. Hofmann, 2004. Hierarchical document categorization with support vector machines. In Proceedings of the Proceedings of the thirteenth ACM international conference on Information and knowledge management

(Washington, D.C., USA2004), ACM, 1031186, 78-87. DOI= http://dx.doi.org/10.1145/1031171.1031186.
[5] I. Brown and C. Mues, 2012. An experimental comparison of classification algorithms for imbalanced credit scoring data sets. Expert Systems with Applications 39, 3, 3446-3453.
[6] P. C. Pendharkar, 2005. A threshold-varying artificial neural network approach for classification and its application to bankruptcy prediction problem. Computers & Operations Research 32, 10 (2005/10/01/), 2561-2582. DOI= http://dx.doi.org/https://doi.org/10.1016/j.cor.2004.06.023.
[7] C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel, 2008. Decision trees for hierarchical multi-label classification. Machine Learning 73, 2 (August 01), 185. DOI= http://dx.doi.org/10.1007/s10994-008-5077-3.
[8] J. Ham, C. Yangchi, M. M. Crawford, and J. Ghosh, 2005. Investigation of the random forest framework for classification of hyperspectral data. IEEE Transactions on Geoscience and Remote Sensing 43, 3, 492-501. DOI= http://dx.doi.org/10.1109/TGRS.2004.842481.
[9] A. M. Cruz, 2013. Evaluating record history of medical devices using association discovery and clustering techniques. Expert Systems with Applications 40, 13, 5292-5305.
[10] R. Bhattacharyya, 2011. Cohesion: A concept and framework for confident association discovery with potential application in microarray mining. Applied Soft Computing 11, 1 (2011/01/01/), 592-604. DOI= http://dx.doi.org/https://doi.org/10.1016/j.asoc.2009.12.018.
[11] K. A. Nazeer and M. Sebastian, 2009. Improving the Accuracy and Efficiency of the k-means Clustering Algorithm. In Proceedings of the world congress on engineering Association of Engineers London, 1-3.
[12] C. a. C. Coello, G. B. Lamont, and D. A. Van Veldhuizen, 2007. Evolutionary algorithms for solving multi-objective problems. Springer.
[13] R. Poli, W. B. Langdon, N. F. Mcphee, and J. R. Koza, 2008. A field guide to genetic programming. Lulu. com.
[14] Z. Mengjie, G. Xiaoying, and L. Weijun, 2006. Looseness Controlled Crossover in GP for Object Recognition. In 2006 IEEE International Conference on Evolutionary Computation, 1285-1292. DOI= http://dx.doi.org/10.1109/CEC.2006.1688457.
[15] A. Song, V. Ciesielski, and H. E. Williams, 2002. Texture classifiers generated by genetic programming. In Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600), 243-248 vol.241. DOI= http://dx.doi.org/10.1109/CEC.2002.1006241.
[16] M. Zhang, P. Andreae, and M. Pritchard, 2003. Pixel Statistics and False Alarm Area in Genetic Programming for Object Detection. In Applications of Evolutionary Computing, S. CAGNONI, C.G. JOHNSON, J.J.R. CARDALDA, E. MARCHIORI, D.W. CORNE, J.-A. MEYER, J. GOTTLIEB, M. MIDDENDORF, A. GUILLOT, G.R. RAIDL and E. HART Eds. Springer Berlin Heidelberg, Berlin, Heidelberg, 455-466.
[17] M. Zhang, V. B. Ciesielski, and P. Andreae, 2003. A Domain-Independent Window Approach to Multiclass Object Detection Using Genetic Programming. EURASIP Journal on Advances in Signal Processing 2003, 8 (July 21), 206791. DOI= http://dx.doi.org/10.1155/s1110865703303063.
[18] M. Zhang and W. Smart, 2006. Using Gaussian distribution to construct fitness functions in genetic programming for multiclass object classification. Pattern Recognition Letters 27, 11 (2006/08/01/), 1266-1274. DOI= http://dx.doi.org/https://doi.org/10.1016/j.patrec.2005.07.024.
[19] K.-H. Liu and C.-G. Xu, 2008. A genetic programming-based approach to the classification of multiclass microarray datasets. Bioinformatics 25, 3, 331-337. DOI= http://dx.doi.org/10.1093/bioinformatics/btn644.
[20] M. Chabbouh, S. Bechikh, C.-C. Hung, and L. Ben Said, 2019. Multi-objective evolution of oblique decision trees for imbalanced data binary classification. Swarm and Evolutionary Computation 49(2019/09/01/), 1-22. DOI= http://dx.doi.org/https://doi.org/10.1016/j.swevo.2019.05.005.
[21] S. Bechikh, M. Elarbi, C.-C. Hung, S. Hamdi, and L. B. Said, 2019. A Hybrid Evolutionary Algorithm with Heuristic Mutation for Multi-objective Bi-clustering. In 2019 IEEE Congress on Evolutionary Computation (CEC) IEEE, 2323-2330.
[22] M. Hammami, S. Bechikh, C.-C. Hung, and L. B. Said, 2019. Weighted-Features Construction as a Bi-level Problem. In 2019 IEEE Congress on Evolutionary Computation (CEC) IEEE, 1604-1611.
[23] A. Tahmassebi, A. H. Gandomi, I. Mccann, M. H. Schulte, L. Schmaal, A. E. Goudriaan, and A. Meyer-Baese, 2017. An evolutionary approach for fMRI big data classification. In 2017 IEEE Congress on Evolutionary Computation (CEC), 1029-1036. DOI= http://dx.doi.org/10.1109/CEC.2017.7969421.
[24] H. Al-Sahaf, M. Zhang, M. Johnston, and B. Verma, 2015. Image descriptor: A genetic programming approach to multiclass texture classification. In 2015 IEEE Congress on Evolutionary Computation (CEC), 2460-2467. DOI= http://dx.doi.org/10.1109/CEC.2015.7257190.
[25] J. R. Koza, 1992. Genetic programming: on the programming of computers by means of natural selection. MIT Press.
[26] W. Banzhaf, J. R. Koza, C. Ryan, L. Spector, and C. Jacob, 2000. Genetic programming. IEEE Intelligent Systems and their Applications 15, 3, 74-84. DOI= http://dx.doi.org/10.1109/5254.846288.
[27] R. Poli, W. B. Langdon, and N. F. Mcphee, 2008. A Field Guide to Genetic Programming. Lulu Enterprises, UK Ltd.
[28] B. Tran, B. Xue, and M. Zhang, 2019. Genetic programming for multiple-feature construction on high-dimensional classification. Pattern Recognition 93, 404-417.
[29] S. Picek, E. Hemberg, D. Jakobovic, and U.-M. O'reilly, 2018. One-Class Classification of Low Volume DoS Attacks with Genetic Programming. In Genetic Programming Theory and Practice XV Springer, 149-168.
[30] J. Madár, J. Abonyi, and F. Szeifert, 2005. Genetic Programming for the Identification of Nonlinear Input–Output Models. Industrial & Engineering Chemistry Research 44, 9 (2005/04/01), 3178-3186. DOI= http://dx.doi.org/10.1021/ie049626e.
[31] P. Płoński and K. Zaremba, 2014. Visualizing Random Forest with Self-Organising Map Springer International Publishing, Cham, 63-71.
[32] N. Manwani and P. S. Sastry, 2012. Geometric Decision Tree. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42, 1, 181-192. DOI= http://dx.doi.org/10.1109/TSMCB.2011.2163392.