An Evolutionary Framework for Real-Time Fraudulent Credit Detection

Behshad Mohebali
Department of Scientific Computing
Florida State University
Tallahassee, FL, USA
bmohebali@fsu.edu

Gelareh Karbaschi
Department of Computer Information
Purdue University Northwest Hammond
Hammond, IN, USA
gkarbasc@pnw.edu

Amirhessam Tahmassebi
Department of Scientific Computing
Florida State University
Tallahassee, FL, USA
atahmassebi@fsu.edu

Anke Meyer-Baese
Department of Scientific Computing
Florida State University
Tallahassee, FL, USA
ameyerbaese@fsu.edu

Amir H. Gandomi
Faculty of Engineering and Information Technology
University of Technology Sydney
Ultimo, Australia
gandomi@uts.edu.au

***Abstract*—Fraud has been a worldwide issue that is facing the major economies of the world. Within an economical system, undetected and unpunished fraudulent activities can erode the public trust in law enforcement institutions and even incentivize more fraud. Therefore, detection of fraudulent activities and prosecution of responsible entities is of utmost importance for financial regulatory bodies around the globe. Of the challenges rising with this task is the scarcity of detection resources (auditors) and the fraudsters constantly adapting to the new circumstances of the market. To address these issues, this paper proposes an evolutionary framework for credit fraud detection with the ability to incorporate (and adapt to) the incoming data in real-time. The goal of the framework is to identify the entities with high a risk of fraud for efficient targeting of the scarce resources. The data that is generated as a result of the audits are fed into the framework for further training.**

I. Introduction

As the use of credit card as a major means of payment for customers becomes evermore prevalent, the number of fraudulent transactions increases as well [1]. Besides the monetary cost of fraud, the erosion of trust within a market can result in more regulatory restrictions, higher transaction costs, and less efficiency in the market [2]. Therefore, it is crucial to detect fraud efficiently and quickly to maintain the integrity of the market [3]. As technology has made transactions easier to make, fraud detection authorities are faced with new challenges due to the size of the activities in need of supervision. The traditional methods of fraud detection that involve manual detection have been rendered less and less practical with the emergence of big data technologies [1], [4], [5]. At the same time, the abundance of data presents opportunities for research and development in statistical and mathematical approaches to automate the process of fraud detection.

The use of artificial intelligence (AI) methods in pattern recognition has been growing rapidly in the past decade as the data needed for training machine learning models have become readily available [3]. This has provided researchers and developers with efficient and standard tools to build AI pipelines to detect fraud, especially in credit card transactions [1]. The problem of credit card fraud detection is defined as a binary classification problem [6], with the possible classes for a given transaction being "fraudulent" and "not fraudulent." Given this setting, many approaches have been used to tackle this problem such as artificial neural networks (ANN) [7], [8], [9], genetic algorithm (GA) [10], Support Vector Machines (SVM) [11], decision trees [12], optimization algorithms based on migrating birds [13], logistic regression [14], [15], Bayesian networks (BN) [6] and naive Bayes (NB) [15], [1], and probabilistic neural networks (PNN) [16], [17], to name a few.

In [3] authors make a comparison between ten different classification methods, including PNN, SVM, ANN, NB, BN, etc. The performance of these classifiers were tested by ten performance metrics including type-I and type-II error, accuracy, F1 and F2 scores, Matthew's correlation coefficient (MCC), and Areu under ROC curve. In [1] the comparison is made between NB, K-nearest neighbors, and logistic regression on highly skewed data. The performance metrics used are accuracy, MCC, precision, sensitivity, and specificity of the classifiers. [14] compares decision trees, ANN, and logistic regression and shows that the latter two outperform decision trees. On the other hand, [18] compares SVM, random forests, and logistic regression in detecting credit card fraud. It is stated that it cannot be a good performance metric for unbalanced data accuracy. Therefore, sensitivity, specificity, AUC, G-mean, and F-score were used alongside accuracy to have a better comparison between classifiers [19], [20].

Evolutionary algorithms (EA) in computational intelligence are methods that use natural selection to achieve "machine intelligence" [21], [22]. Genetic algorithm [23] applies this principle to the problem of search while Genetic Programming (GP) extends that to the domain of computer programs [24], [25], [26], [27]. GP is a heuristic evolutionary method that

978-1-7281-8393-0/21/$31.00

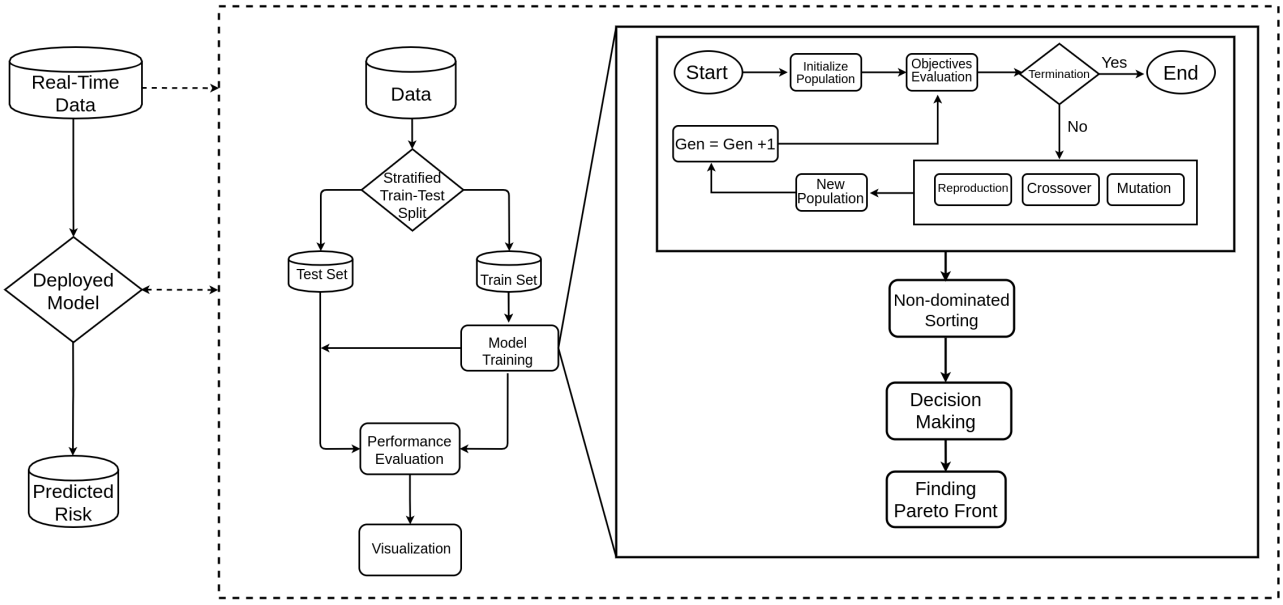


Fig. 1. Flowchart of the evolutionary framework.

uses symbols in complex patterns in search of a function representation [28], which has been used in classification applications extensively [29]. It is noteworthy that the use of GP for classification offers advantages such as flexibility to the type of problems, implicit feature extraction, and good interpretability of the discriminant function. The simplest form of classification is binary classification [30], [31], [32]. In [33] evolutionary algorithms were applied to the specific task of fraud detection formulated as a classification problem.

In [34] three active learning methods were compared with the purpose of applying GP to large datasets. The idea of active learning is that a machine learning algorithm can achieve higher performance with less data if it can choose the data from which it learns [35]. Given that training on large datasets for evolutionary algorithms is time consuming and in some cases to the point of being prohibitive, [34] uses a query strategy to limit the size of the training without sacrificing performance. The process of query selection is done while the GP process is running.

There are several aspects of the credit card fraud detection data that is of interest: First, it is constantly being generated as long as people use credit card as a means of payment. Second, the methods of fraud is always evolving and fraudsters are being more sophisticated with their methods. This paper proposes a GP based classification framework that can take advantage of the constant stream of data using a real-time training approach that updates the classification model in episodes.

The remainder of the paper is structured as follows. Section II discusses the real-time evolutionary framework developed to detect credit card fraud. Section III briefly explains the internals of a Genetic Programming-based classifier. Section IV describes the data used in this work based on [3]. Section V presents the results from the experiment using the framework and the data. And finally, Section VI concludes the paper while proposing future directions for this research.

II. Evolutionary Real-Time Framework

The proposed real-time evolutionary framework has four main components:

1) Extract, transform, load (ETL) process
2) Training
3) Deployment
4) Prediction

The framework schematic is shown in Fig. 2.The process starts with loading the real-time data and ETL process. The model is trained and deployed for prediction of the risk for fraudulent activities. In this part, we neglect the challenges involved in the ETL process whether the data lives in the cloud or in any other kind of server. Here, we assume that the data has been loaded into the memory and we can focus more on the modeling aspect of the framework. The dashed square in the flowchart presents the main training phase of the framework. As seen, the in-memory data is being splitted into train/test splits in a stratified fashion to keep the prevalence of both populations the same. This also helps us down the road when we deal with imbalanced classification problems to generalize the problem with better confidence [36].

In the training step, the feature engineering, feature standardization, feature scaling, and feature encoding can be fitted to the train data. Then, the trained object for each step can be used to transform the test set. Next, the probability assigned to each operation should be tuned. This could be done in an ad-hoc fashion to figure out the type of functions or the rate of crossover or mutation in the MOGP classifier. In addition to this, other hyper-parameter tuning fashions

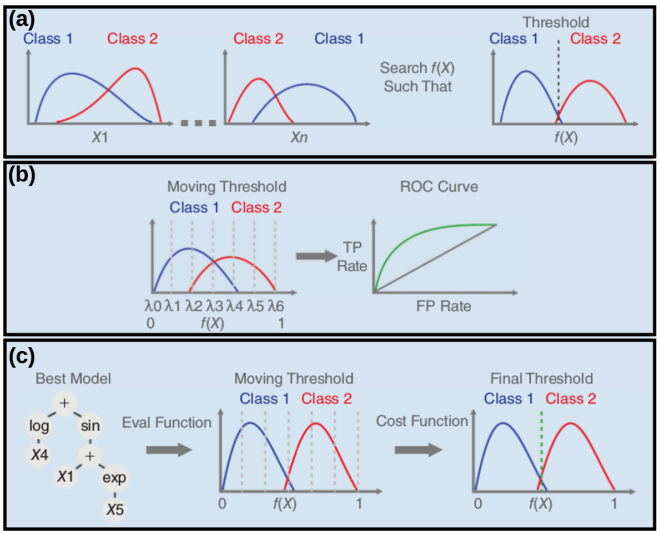


Fig. 2. GP function classifier: (a) Search, (b) Evaluation, (c) Threshold selection[39][40].

including exhaustive grid-search, random-search, and Bayesian optimization can be used [37]. The model training is shown on the right-hand side of the flowchart where the MOGP algorithm is being implemented and the results go through the non-dominated sorting algorithm as we discussed in Algorithm 1 and the Pareto Front of the solution is being developed. Once the training process is done, the trained model is applied on the test set and the common performance metrics for classification problems such as sensitivity, specificity, precision, and recall are calculated [38].

Last, the visualization module would be applied to the results for both training and testing sets. Now, we have a trained model which has been validated via the test set. Therefore, the model can be re-fit to the whole in-memory data. The retrained model can be deployed into the production to be ready for any real-time data. Each real-time data would be fed into the deployed model and the risk of fraudulent credit would be calculated. As shown, the training phase has a feedback loop to the deployment module. Therefore, the real-time data will be stored in the temporary table and will be joined back to the training phase.

The most common approach for real-time risk prediction is batch-processing. The predicted data can be used alongside the training data. After the model is trained on the combination of the training data and the predicted data, the new version of the model will be deployed. The frequency of this retraining scheme depends on the use-case and the resources available to the framework [36].

III. GP Function Classifier

Genetic Programming (GP) is formulated as symbolic optimization problem due to the selection of designs applied to the fitness and complexity metrics phase. It was originally an evolutionary approach based on the principles of Darwinian natural selection [41], [29], [42], [43]. The GP classifier

Algorithm 1: NSGA-II

Input: Generations N, Population P
Output: Best Model

```
Initialize population P;
Generate random population size M;
Evaluate objective values;
Assign ranking based on Pareto sort;
Generate child population;
Binary tournament selection;
Recombination and mutation;
for i ∈ {1, ..., N} do
    for each Parent and Child in Population do
        Assign ranking based on Pareto sort;
        Generate sets of non-dominated solutions;
        Determine Crowding distance;
        Loop inside by adding solutions to next
          generation starting from the first front until N
          individuals;
    end
    Select points on the lower front with higher
      crowding distance;
    Create next generation;
    Binary tournament selection;
    Recombination and mutation;
end
```

obtains the label of a data point by calculating a real value based on each symbol tree and comparing it to its set threshold. GP training algorithm uses a group of individuals (called the population) and genetic operators to create new set of individuals (generations) based on the metrics that determine the quality of each individual, namely the fitness and complexity functions.

TABLE I
EXTRACTED FEATURES FOR THE GP CLASSIFIER.

Feature	Variable	Description
Sector Score	X_1	Historical risk score of the sector
Parameter A	X_2	Discrepancy found in report A
Score A	X_3	
Parameter B	X_4	Discrepancy found in report B
Score B	X_5	
Total	X_6	Total discrepancy found in other reports
Numbers	X_7	Historical discrepancy score
Marks	X_8	
Money Value	X_9	amount of money involved
Money Marks	X_{10}	
Loss	X_{11}	Amount of loss suffered by the firm last year
Loss Score	X_{12}	
History	X_{13}	Average historical loss in the last 10 years
History Score	X_{14}	
Location ID	X_{15}	Unique ID of the location
District	X_{16}	

The GP classifier as shown in Fig. 2, [39] was implemented as a multi-objective GP system based on the non-dominant sorting genetic algorithm II [44] (NSGA-II). The details of NSGA-II are explained in Algorithm 1. GP binary classifier training process includes finding a non-linear function $y = f(\bar{X})$ such that for two classes of H_0 and H_1, the two distributions of $p(f(\bar{X})|H_0)$ and $p(f(\bar{X})|H_1)$ are best separated, as shown in Fig. 2a. The predicted label of each input X_i is determined by passing the output of each learning function through the decision rule $\hat{L}_i$ defined as:

$$\hat{L}_i = \begin{cases} 0 & \text{if } y_i \geq \lambda \\ 1 & \text{if } y_i < \lambda \end{cases} \quad (1)$$

where $0 \leq \lambda \leq 1$ is the threshold that separates two classes H_0 and H_1 for input X_i based on $y_i = f(X_i)$. The highest discriminatory power for the two classes can be obtained by choosing the threshold that results in the largest value of area under the Receiver Operating Characteristics (ROC) curve, as shown in Fig. 2b. The change in λ can cause a change in classification error, false positive rate (FPR), and false negative rate. To obtain the best λ, an optimization problem was defined with the weighted sum of the false positive (FP) and false negative rates as the cost function. The threshold λ was calculated by minimizing the cost function, as shown in Figure 2c. The sub-tree complexity measure (CM) [45] was considered as the second objective of the optimization process.

IV. Data

The collected data is a one year non-confidential data set between 2015 and 2016 from 777 firms in 46 cities by Auditor General Office (AGO) of Comptroller and Auditor General (CAG) of India [3]. These firms span across 14 sectors of the Indian economy. A total of sixteen features, $\{X_1, \cdots, X_{16}\}$, as shown in Table I were extracted for the classification task. In addition to this, a set of binary labels (fraudulent or not-fraudulent) was used for each sample. The main features of the data were selected through close collaboration with the auditors. The label for each instance of the data is determined through calculating the Audit risk Score and a threshold to separate "Fraud" firms from "Non-fraud" ones. This score can be formulated as the combination of several possible risk types into one risk measure, for example in the form of a summation. Fig. 3 depicts the lower triangle Pearson correlation matrix of the extracted features with a color bar (the lighter color, the more correlation) which highlights the probability of the correlation of each of the extracted features with each other. As expected, the total discrepancy and parameter B (the discrepancy in report B), and the loss score and total loss in the first year are correlated, while other features hold correlation coefficients of less than 0.6.

V. Results & Discussion

To see the performance of the proposed framework, an MOGP model based on the NSGA-II algorithm was developed. A total of sixteen extracted features were used as the inputs for the detection of the fraudulent credits. An ad-hoc

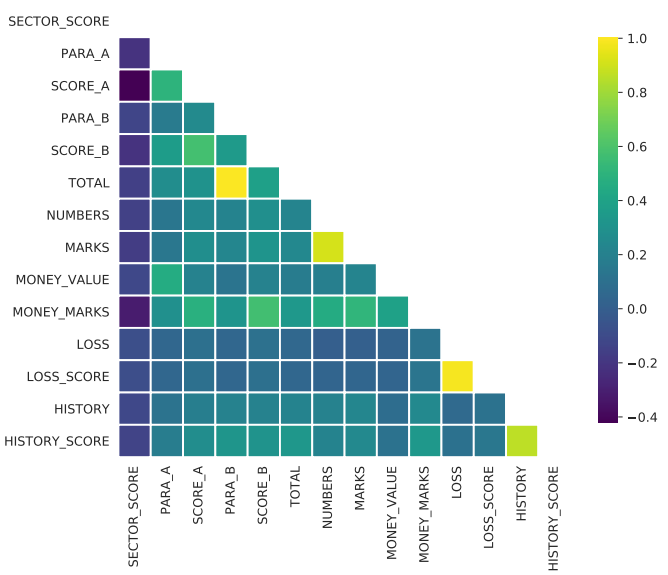


Fig. 3. Correlation matrix plot of the features.

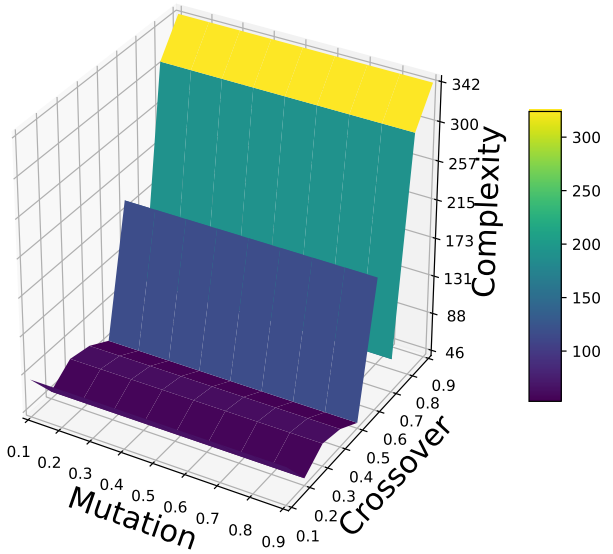


Fig. 4. 3D surface plot of the MRs and the CRs versus FM.

trial (a model trained over 100 generations) for the mutation and the CRs of the model for various rates ranging from 0.1 to 0.9 were employed. Fig. 4 illustrates the three-dimensional surface plot of the MRs and the CRs based on the fitness measure (FM). The fitness of the model was converged fast; therefore, complexity was chosen here as the metric to find the best crossover and MR for the main run. The lowest complexity value was reported with 0.7 as the CR and 0.3 as for the MR. The resultant measures were used to develop the final MOGP model evolved through 2000 generations and 1000 populations. The details of the parameters setting for the GP classifier are presented in Table II.

TABLE II
PARAMETERS SETTING FOR THE GP FUNCTION CLASSIFIER.

Parameter	Setting
Population Size	1000
Number of Generations	2000
Tournament Size	20
Number of Inputs	16
Crossover Rate (CR)	0.7
Mutation Rate (MR)	0.3
Number of CPU Threads	8
1^{st} Objective	AUC
2^{nd} Objective	Sub-tree Complexity
Population Initialization	Ramped-Half-and-Half
Function Set	$+, -, \times, /, \sqrt{\ }, (\ )^2, (\ )^3, (\ )^4$ $\log, \exp, \sin, \cos$

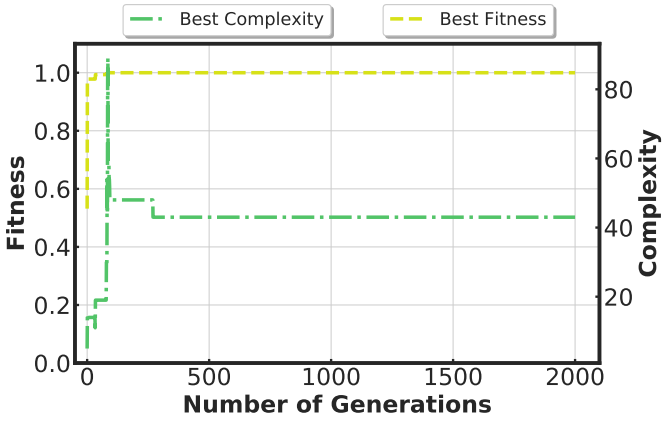


Fig. 5. GP evolution through generations. Left axis and right axis present fitness and complexity, respectively.

The evolution of the model training can be shown as the variation of the objectives of the model, fitness and complexity at each generation as shown in Fig. 5. The left Y-axis was set to FM, the right Y-axis to complexity, and X-axis to the number of generations. It was observed that after 100 generations the model reached the highest complexity (86) and a fitness (0.99) at the same time. However, as the number of generations increased, the CM decreased by around half. After 2000 generations the complexity reached a value of 43, approximately half of the highest value of CM reached by this model. Additionally, the FM stayed constant through the number of generations.

The evolutionary aspect of the proposed framework opens new avenues to define multiple objective-dependent models. For example, by looking at the first generation of the evolution, the least complex model can be defined with sub-tree complexity of 5 and fitness score of 0.531844. As seen in Fig. 5, after generation 86, both fitness score and sub-tree complexity values converge to 1.0 and 59, respectively. This point is being referred as "knee" where the rate of score improvement for the objectives (fitness and sub-tree complexity) does not get

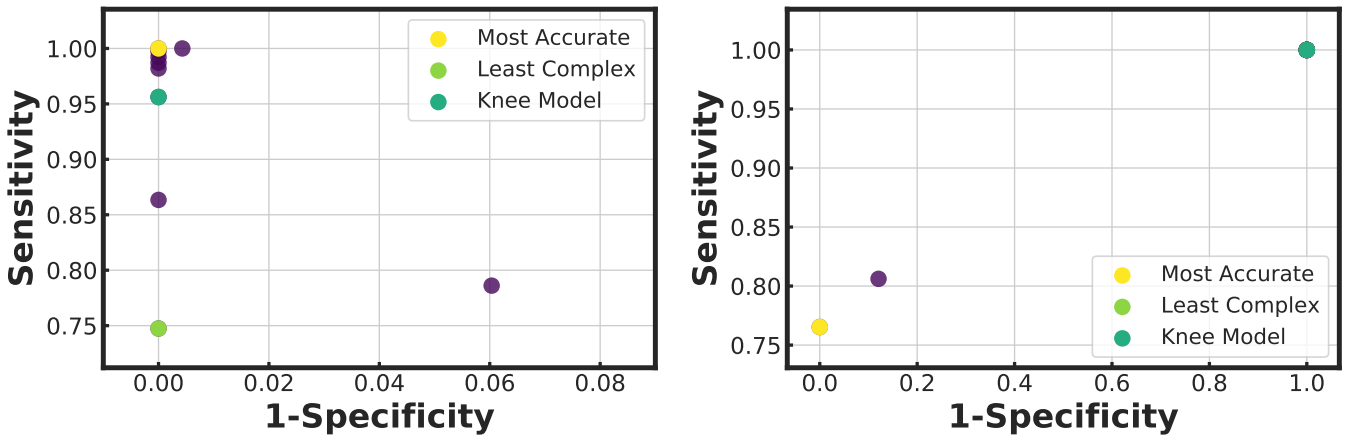
TABLE III
SET OF SYMBOLIC GP SOLUTIONS IN THE PARETO FRONT.

Index	GP Functions
1	X_3
2	$\sin X_2$
3	$\frac{X_3+X_5}{X_{16}}$
4	$\frac{X_3}{X_{16}}$
5	$\frac{X_3+X_5+X_{16}}{X_{16}}$
6	$\frac{X_3+X_5+X_{10}}{X_{16}(X_7-X_{12}-X_{14})}$
7	$\frac{X_3+X_5+X_{10}}{X_{14}+X_{16}}$
8	$\frac{X_3+X_5}{X_9+X_{16}}$
9	$\frac{X_3+X_5}{(X_9+X_{16})(X_7-X_{12}-X_{14})}$
10	$\frac{X_3+X_5+X_{10}}{X_{11}-X_{14}+X_{16}}$

better during the evolution of the model. Last, the obtained best fitness score (1.0) with sub-tree complexity of 89 after 85 generations can be used to define the most accurate model. As shown in previous studies [40], the performance of the most accurate model and the knee model are usually similar while the computational run-time of them can vary depending on the number of generations and populations. However, in this study, we can see that the sub-tree complexity drops at the knee point.

As discussed, the resulted Pareto front of the developed MOGP models based on Algorithm 1 comprises 10 models. The symbolic representation of each model in the Pareto front is shown in Table III. The symbolic representation would help the fraud analysts to employ various sensitivity analysis using synthetic data to see what features would need more attention for the development of a better fraud detection classifier. That could be the basis of feature importance of the developed model. For instance, Score A (X_3) is the most frequent feature (90% total cover) across all of the models in the Pareto front. In this case, more engineered features with respect to the feature importance analysis would boost the validation results. Fig. 6 and Fig. 7 illustrate the trade-off of sensitivity-specificity and precision-recall of all developed models resulted in the Pareto front for both training and testing data sets. The Pareto front models are presented in dark purple circles, the most accurate models are presented in yellow circles, the least complex models are shown in light green circles, and the knee models are shown in sea green circles.

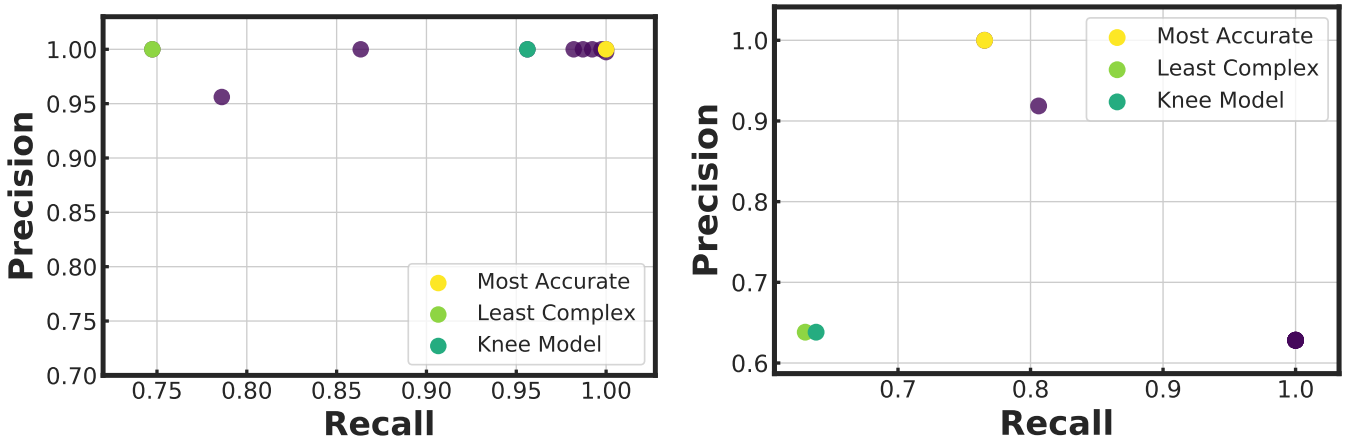
Sensitivity is defined as the proportion of the cases correctly classified as positive to the total number of positive cases. It is also known as true positive rate, or recall. Specificity, on the other hand, is the proportion of the cases correctly classified as negative to the total number of the negative cases. The performance of the classifier can be evaluated comparing these two



(a) Training

(b) Testing

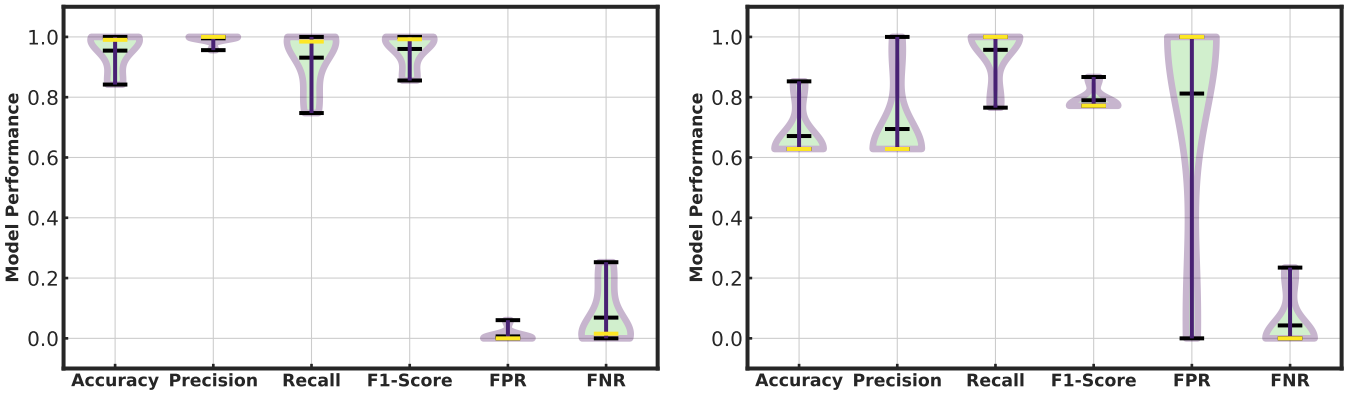
Fig. 6. Scatter plots of (a) training and (b) testing phases of sensitivity-specificity of all models in the Pareto front (dark purple circles) with specifying the most accurate model (yellow circle), the least complex model (light green circle) and the knee model (sea green circle).



(a) Training

(b) Testing

Fig. 7. Scatter plots of (a) training and (b) testing phases of precision-recall of all models in the Pareto front (dark purple circles) with specifying the most accurate model (yellow circle), the least complex model (light green circle) and the knee model (sea green circle).



(a) Training

(b) Testing

Fig. 8. Violin plots of the classification performance metrics of all the models in the Pareto front for (a) training and (b) testing. Minimum, maximum, and mean values are presented with horizontal black lines. Yellow line indicates the median for each metric.

metrics: how well the classifier separates the negative cases from the positive cases. The ideal case is higher sensitivity and high specificity. As specificity is between 0 and 1, 1-specificity stays in the same bounds. Precision is the ratio of the number of correct positive predictions to the number of all the cases predicted positive (either truly or falsely).

Furthermore, Fig. 8 illustrates the distribution of the various classification metrics including accuracy, precision, recall, F1-score, FPR, and false negative rate (FNR) of all of the models in the resulted Pareto front and their probability density estimation for both training and testing data sets. A violin plot was used indicating the minimum, maximum, mean, and median values for each metric. As seen, models in the Pareto front have shown reasonable results ($\pm 10\%$) based on the available data in this study. It should be noted that the training and testing data are stratified to make sure the prevalence of fraud in both populations is the same. Therefore, the real-time validation engine would be more realistic. However, the results presented by Hooda et al. [3] are based on non-stratified 10-folds cross-validation. This would be the reason that we have a variation in FPR of the model predictions of the testing data set while the FPR of the training data is negligible.

VI. Conclusions & Future Work

This paper proposes an evolutionary framework for real-time fraudulent credit detection. In this regard, an MOGP model based on NSGA-II algorithm was developed. In this model, the area under the ROC curve and the sub-tree complexity were used as the objectives. The MPGP model ran for 2000 generations with 1000 populations. Additionally, the mutation and CRs were optimized over 100 generations based on the sub-tree CMs. A total of sixteen extracted features were utilized as the inputs of the MOGP classifier. Table III presents the resulted symbolic GP solutions in the Pareto front. The classification metrics of the resultant models in Pareto front as seen in Fig. 8, shown reasonable performance for our stratified training/testing data model. After execution of the MOGP process, the optimal model was selected from the Pareto front results. This was done by a trade-off between the fitness and the model complexity. Based on the results, the following conclusions are drawn:

- During the training evolution, the model quickly reached to a high fitness score. Therefore, the sub-tree complexity was used as the main objective to choose the genetic operators probability as shown in Fig. 4.
- The Pareto front includes 10 models as shown in Table III where the least complex model is $F(X) = X_3$ and the most complex model is $F(X) = \frac{X_3+X_5}{(X_9+X_{16})(X_7-X_{12}-X_{14})}$.
- Based on the resulted models in Pareto front, discrepancy score based on report A (X_3), district (X_{16}), and discrepancy score based on report B (X_5) are the top three important features with total cover of 90%, 80%, and 70%, respectively. These features can be analyzed in future studies to generate more salient features to boost the fraudulent cases and reduce the FPR.
- The FPR resultant has shown a high variance in the testing phase while having a negligible variance in the training phase. This could be due to the data limitation, or the random combination of the stratified fashion of training/testing. In the future studies, the framework can be retrained using a bigger dataset to decrease the variance of the FP cases in the model predictions.
- The proposed MOGP framework is shown to be a fast enough tool that can be used to tackle big data problems to generate solid symbolic solutions. Symbolic solutions would give credit analysts the ability to employ various sensitivity analysis to interpret the results based on the employed features. That would open up new insights to improve the fraudulent credit detection by designing new non-linear criteria and meta-heuristic rules as new features.

Acknowledgments

The authors would like to thank Mahta Zakaria for the careful revision of the manuscript.

References

[1] J. O. Awoyemi, A. O. Adetunmbi, and S. A. Oluwadare, "Credit card fraud detection using machine learning techniques: A comparative analysis," in *2017 International Conference on Computing Networking and Informatics (ICCNI)*. IEEE, 2017, pp. 1–9.

[2] J. Perols, "Financial statement fraud detection: An analysis of statistical and machine learning algorithms," *Auditing: A Journal of Practice & Theory*, vol. 30, no. 2, pp. 19–50, 2011.

[3] N. Hooda, S. Bawa, and P. S. Rana, "Fraudulent firm classification: a case study of an external audit," *Applied Artificial Intelligence*, vol. 32, no. 1, pp. 48–64, 2018.

[4] B. Mohebali, A. Tahmassebi, A. H. Gandomi, and A. Meyer-Baese, "A big data inspired preprocessing scheme for bandwidth use optimization in smart cities applications using raspberry pi," in *Big Data: Learning, Analytics, and Applications*, vol. 10989. International Society for Optics and Photonics, 2019, p. 1098902.

[5] A. Tahmassebi, A. Ehtemami, B. Mohebali, A. H. Gandomi, K. Pinker, and A. Meyer-Baese, "Big data analytics in medical imaging using deep learning," in *Big Data: Learning, Analytics, and Applications*, vol. 10989. International Society for Optics and Photonics, 2019, p. 109890E.

[6] S. Maes, K. Tuyls, B. Vanschoenwinkel, and B. Manderick, "Credit card fraud detection using bayesian and neural networks," in *Proceedings of the 1st international naiso congress on neuro fuzzy technologies*, 2002, pp. 261–270.

[7] K. M. Fanning and K. O. Cogger, "Neural network detection of management fraud using published financial data," *Intelligent Systems in Accounting, Finance & Management*, vol. 7, no. 1, pp. 21–41, 1998.

[8] F. N. Ogwueleka, "Data mining application in credit card fraud detection system," *Journal of Engineering Science and Technology*, vol. 6, no. 3, pp. 311–322, 2011.

[9] B. P. Green and J. H. Choi, "Assessing the risk of management fraud through neural network technology," *Auditing*, vol. 16, pp. 14–28, 1997.

[10] K. RamaKalyani and D. UmaDevi, "Fraud detection of credit card payment system by genetic algorithm," *International Journal of Scientific & Engineering Research*, vol. 3, no. 7, pp. 1–6, 2012.

[11] G. Singh, R. Gupta, A. Rastogi, M. D. Chandel, and A. Riyaz, "A machine learning approach for detection of fraud based on svm," *International Journal of Scientific Engineering and Technology*, vol. 1, no. 3, pp. 194–198, 2012.

[12] J. R. Gaikwad, A. B. Deshmane, H. V. Somavanshi, S. V. Patil, and R. A. Badgujar, "Credit card fraud detection using decision tree induction algorithm," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 4, no. 6, 2014.

[13] E. Duman, A. Buyukkaya, and I. Elikucuk, "A novel and successful credit card fraud detection system implemented in a turkish bank," in *2013 IEEE 13th International Conference on Data Mining Workshops*. IEEE, 2013, pp. 162–171.

[14] A. Shen, R. Tong, and Y. Deng, "Application of classification models on credit card fraud detection," in *2007 International conference on service systems and service management*. IEEE, 2007, pp. 1–4.

[15] A. Jordan *et al.*, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," *Advances in neural information processing systems*, vol. 14, no. 2002, p. 841, 2002.

[16] P. Ravisankar, V. Ravi, G. R. Rao, and I. Bose, "Detection of financial statement fraud and feature selection using data mining techniques," *Decision support systems*, vol. 50, no. 2, pp. 491–500, 2011.

[17] B. Mohebali, A. Tahmassebi, A. Meyer-Baese, and A. H. Gandomi, "Probabilistic neural networks: a brief overview of theory, implementation, and application," *Handbook of Probabilistic Models*, pp. 347–367, 2020.

[18] S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, "Data mining for credit card fraud: A comparative study," *Decision support systems*, vol. 50, no. 3, pp. 602–613, 2011.

[19] K. Jiang, T. Chen, L. Huang, G. Karbaschi, and G. R. Bernard, "Semantic similarity of side effect and indication relations of drugs inferred from neural embedding." in *SEPDA@ ISWC*, 2019, pp. 68–77.

[20] K. Jiang, L. Huang, T. Chen, G. Karbaschi, D. Zhang, and G. R. Bernard, "Mining potentially unreported effects from twitter posts through relational similarity: A case for opioids," in *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2020, pp. 2603–2609.

[21] J. R. Koza, *Genetic programming: on the programming of computers by means of natural selection*. MIT press, 1992, vol. 1.

[22] A. M. Turing, "Computing machinery and intelligence," in *Parsing the turing test*. Springer, 2009, pp. 23–65.

[23] J. H. Holland *et al.*, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[24] J. R. Koza, "Human-competitive results produced by genetic programming," *Genetic programming and evolvable machines*, vol. 11, no. 3-4, pp. 251–284, 2010.

[25] W. Banzhaf, L. Spector, and L. Sheneman, *Genetic Programming Theory and Practice XVI*. Springer, 2019.

[26] A. Tahmassebi and A. H. Gandomi, "Genetic programming based on error decomposition: A big data approach," in *Genetic programming theory and practice XV*. Springer, 2018, pp. 135–147.

[27] ——, "Building energy consumption forecast using multi-objective genetic programming," *Measurement*, vol. 118, pp. 164–171, 2018.

[28] W. Banzhaf, J. Koza, C. Ryan, L. Spector, and C. Jacob, "Genetic programming," *IEEE Intelligent Systems and their Applications*, vol. 15, no. 3, pp. 74–84, 2000.

[29] P. G. Espejo, S. Ventura, and F. Herrera, "A survey on the application of genetic programming to classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 2, pp. 121–144, 2010.

[30] W. A. Tackett, "Genetic programming for feature discovery and image discrimination." in *ICGA*. Citeseer, 1993, pp. 303–311.

[31] D. Hope, E. Munday, and S. Smith, "Evolutionary algorithms in the classification of mammograms," in *2007 IEEE Symposium on Computational Intelligence in Image and Signal Processing*. IEEE, 2007, pp. 258–265.

[32] G. Wilson and M. Heywood, "Introducing probabilistic adaptive mapping developmental genetic programming with redundant mappings," *Genetic Programming and Evolvable Machines*, vol. 8, no. 2, pp. 187–220, 2007.

[33] A. Pouramirarsalani, M. Khalilian, and A. Nikravanshalmani, "Fraud detection in e-banking by using the hybrid feature selection and evolutionary algorithms," *IJCSNS*, vol. 17, no. 8, pp. 271–279, 2017.

[34] R. Curry, P. Lichodzijewski, and M. I. Heywood, "Scaling genetic programming to large datasets using hierarchical dynamic subset selection," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 4, pp. 1065–1073, 2007.

[35] B. Settles, "Active learning literature survey," 2009.

[36] A. Tahmassebi, J. Martin, A. Meyer-Baese, and A. H. Gandomi, "An interpretable deep learning framework for health monitoring systems: A case study of eye state detection using eeg signals," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2020, pp. 211–218.

[37] A. Tahmassebi, "ideeple: Deep learning in a flash," in *Disruptive Technologies in Information Sciences*, vol. 10652. International Society for Optics and Photonics, 2018.

[38] A. Tahmassebi, A. H. Gandomi, and A. Meyer-Baese, "A pareto front based evolutionary model for airfoil self-noise prediction," in *2018 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2018, pp. 1–8.

[39] I. Arnaldo, K. Veeramachaneni, A. Song, and U.-M. O'Reilly, "Bring your own learner: A cloud-based, data-parallel commons for machine learning," *IEEE Computational Intelligence Magazine*, vol. 10, no. 1, pp. 20–32, 2015.

[40] A. Tahmassebi, A. H. Gandomi, and A. Meyer-Baese, "An evolutionary online framework for mooc performance using eeg data," in *2018 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2018, pp. 1–8.

[41] T. Loveard and V. Ciesielski, "Representing classification problems in genetic programming," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 2. IEEE, 2001, pp. 1070–1077.

[42] A. H. Gandomi, A. H. Alavi, and C. Ryan, *Handbook of genetic programming applications*. Springer, 2015.

[43] A. Tahmassebi and T. Smith, "Slickml: Slick machine learning in python," 2021. [Online]. Available: https://github.com/slickml/slick-ml

[44] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[45] E. Y. Vladislavleva *et al.*, *Model-based problem solving through symbolic regression via pareto genetic programming*. CentER, Tilburg University, 2008.