IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834, p-ISSN: 2278-8735
PP 50-56
www.iosrjournals.org

Classification of Imbalanced Data using Fitness Functions

Pundlik A. A.^{1}, Patil B. J.^{2} & Boraste P.P.^{3}
^{1,2,3}*(Comp Engg. Dept., KVNNIEER Nasik,/SPP Univ., Pune(MS) ,India)*

Abstract : *Fitness functions are use to get better accuracy when datasets are Imbalanced i.e .data sets are divided into minority and majority classes. Classifiers have good accuracy on the majority classes but very poor accuracy on the minority classes. There are two approaches to classify Imbalanced data such as external and internal. External approach uses Transforming & Sampling from original unbalanced dataset. Internal approach uses Cost adjustment with the help of fitness function in the uneven distribution of dataset during the training process. The fitness function plays a very important role in Genetic Programming to obtain the best solutions within a large search space more effectively & efficiently. This paper developed fitness functions like Amse, Incr, Corr and Dist with Naive Bayes and Support vector machine methods.*

Keywords - *Fitness Function, Genetic Programming, Imbalanced Data, NB, SVM.*

1. INTRODUCTION

Genetic programming (GP) is an encouraging machine learning and search technique which has been fruitful in constructing reliable classifiers[2]. GP is flexible, heuristic and powerful evolutionary technique with some features that can be very valuable and suitable with greater potential for the evolution of classifiers. Classification is an organized way of predicting class membership for a set of examples or instances using the properties of those examples. In many real-world territories, it is common for data sets to have imbalanced class distributions. This occurs when at least one class is represented by only a small number of examples (called the minority class) while other classes make up the rest (called the majority class). Using an unequal distribution of class examples in the learning process, can leave learning algorithms with a performance bias *i.e.* solutions exhibit high accuracy on the majority classes but poor accuracy on the minority classes. The aim of this work is to provide a genetic model for the evolution of GP classifiers under imbalanced data sets through fitness function. There are various imbalanced data sets like[2] PIMA Indian diabetes data, Ionosphere (Ion), SPECT Heart (Spt),Yeast (Yst1 and Yst2), Pedestrian Images (Ped) and Balance Scale (Bal).

2. FITNESS FUNCTIONS

Fitness function is important concept of genetic programming. The fitness function determines how program is able to solve the problem. Fitness functions designed with a maximum amount of judgment. In general whenever there are multiple inputs for which a program must answer correctly, each input is termed a fitness case. Fitness function is necessary to aggressively bias the distribution of paradigms over which learning is conducted. Fitness function is an objective function that is used to encapsulate, as a single figure of merit, how close a given design solution is to achieve the set aims.

2.1 Confusion Matrix

The accuracy of classifiers must be measured by means of the fitness function. Excellence is based on accuracy, and it is often measured as the ratio between the number of correctly classified examples and the total number of examples. Several metrics for measuring accuracy [3], like precision, support, confidence, recall, sensitivity, specificity, and others, are based on the confusion matrix.

Table I. Confusion Matrix

	Predicted Object	**Predicted non-object**
Actual Object	True positive (TP)	False negative (FN)
Actual non-object	False positive (FP)	True negative (TN)

Where TP are positive instances classified as positive, FN are positive instances classified as negative, FP are negative instances classified as positive, TN are negative instances classified as negative. Based on the confusion matrix, standard GP fitness function for classification can be defined by accuracy [1] as shown in eq.(1) :

$$\text{Acc} = \frac{TP+TN}{TP+TN+FP+FN} \qquad (1)$$

2.2 Existing Fitness functions

This system works on two datasets Pedestrian and Yeast. As pedestrian dataset is an image, first we have to extract statistical features from images, then apply GP to classify data and final step is to implement Fitness functions.

2.2.1 Average class Accuracy (Ave) in Fitness

The function Ave uses a weighted-average classification accuracy of the minority and majority classes in fitness. A_{ve} is explained in eq.(2):

$$\text{Ave} = W \times \left(\frac{TP}{TP+FN}\right) + (1-W) \times \left(\frac{TN}{TN+FP}\right) \qquad (2)$$

Where TP - corresponds to minority accuracy, TN – corresponds to majority accuracy, weighting factor is controlled by W, where $0<W<1$.

2.2.2 AUC in Fitness

The AUC is a useful measure of classification performance that is not dependent on a single class threshold, unlike the traditional accuracy-based measures. Auc has since been widely utilized in the machine learning community to both visualize and measure the performance of a classifier across varying class thresholds. The fitness function Auc_F can be defined using eq.(3):

$$Auc_F = \sum_{i=1}^{N-1} \frac{1}{2}(FP_{i+1} - FP_i)(TP_{i+1} + TP_i) \qquad (3)$$

Where N is the number of class thresholds to be used in the ROC curve, TPi and FPi are the accuracy of the two classes at the class threshold i.

2.2.3 Wilcoxon-Mann-Whitney(Wmw) Fitness

Wmw statistic uses a series of pairwise comparisons between the genetic program outputs, effectively measuring the ordering of minority to majority class outputs. The Wmw statistic as a fitness function Wmw is calculated Using eq.(4) :

$$Wmw = \frac{\sum_{i \in Min} \sum_{j \in Maj} I_{wmw}(p_i, p_j)}{|Min| \times |Maj|} \qquad (4)$$

where Pi and Pj (Should Be Italic) represent the outputs of GP when evaluated on minority and majority classes resp, The indicator function Iwmw returns 1 if Pi >Pj and Pi>=0 or 0 otherwise; This enforces both the zero class threshold and the required ordering of minority and majority class outputs inevolved solutions. The denominator ensures that Wmw returns values between 0 and 1, where 1 indicates optimal Auc and 0 indicates poor Auc.

2.3 New Fitness Functions

In existing system fitness functions are based on only GP. In developing New Fitness functions ,we are using NB and SVM also. New fitness functions considers the magnitude of GP, incremental rewards earned per class, correlation ratio (linear dispersal between two population of data), & distance between class distribution . Naive Bayes[4] classification is a machine-learning technique that can be used to predict to which category a particular data case belongs. In this system NB is use to classify data based on probabilities of correctly classified examples and predictable examples. A Support Vector Machine performs classification by constructing an *N*-dimensional hyperplane that optimally separates the data into two categories[5]. Compared with other standard classifiers, SVM is more accurate on moderately imbalanced data. The reason is that only Support Vectors (SVs) are used for classification and many majority samples far from the decision boundary can be removed without affecting classification. This system studies four new fitness functions.

2.3.1 Amse

This fitness function is based on the Amse, which is a popular machine learning measure for determining the difference between input and output patterns. The goal of this fitness function is to evolve classifiers whose outputs are closely calibrated with the desired or target values for each class, where solutions with smaller deviations between the target and classifier outputs are rewarded with better fitness over solutions with larger differences.

$$Amse = \frac{1}{K}\sum_{c=1}^{K}\left(1 - \frac{\sum_{i=1}^{N_c}(sig(Pci) - Tc)^2}{N_c \times 2}\right) \qquad (5)$$

Where, $sig(x) = \frac{2}{1+e^{-x}} - 1$

Pci represents the output of a GP classifier when evaluated on the ithexample belonging to class c, Nc is the number of examples in class c, and K is the number of classes. The target Tc values for the majority and minority classes are -0.5 and 0.5resp.

2.3.2 Incr

This fitness function improves the traditional Ave by differentiating between solutions which have the same class accuracy, but which use different internal classification models. By counting the average number of incremental rewards earned per class, Incr favors solutions with better classification models.

$$Incr = \frac{1}{K}\sum_{c=1}^{K}\left(\frac{\sum_{j=1}^{Mc}\left[I_{zt}(j, D_{cj}, c).\sum_{i=1}^{N_c} Eq(D_{cj}, P_{ci})\right]}{\frac{1}{2}N_c(N_c+1)}\right) \qquad (6)$$

Where,

$$I_{zt}(r,k,c) = \begin{cases} r, & if\ k \geq 0\ and\ c \in Min \\ & or\ if\ k < 0\ and\ c \in Maj \\ 0, & otherwise \end{cases} \quad Eq(p,q) = \begin{cases} 1, & if\ p = q \\ 0, & otherwise \end{cases}$$

Dcj represents the jth element of the set of distinct genetic program outputs for all examples in class c, and Mc is the number of distinct genetic program outputs for all examples in class c. The denominator corresponds to the maximum reward that a solution can obtain for each class. Eq(6) uses two indicator functions Izt and Eq to calculate the incremental rewards for each class. The first component Izt returns its first argument if the given prediction is correct with respect to the zero class threshold or 0 if otherwise. In this case, j is returned for correct predictions, where *j* is the incremental reward earned for the given prediction. The second component *Eq* returns 1 if two genetic program outputs are the same or 0 if otherwise, this counts the number of different genetic program outputs that evaluate the given value.

2.3.3 Corr

The correlation ratio can be used for classification if we consider the genetic program outputs, when evaluated on the examples from the two classes, as the two populations of data and how these are separated with respect to each other. The higher the dispersal between these two populations, the better the separability of the genetic program outputs for the two classes. The correlation ratio outputs values between 0 and 1.

$$Corr = \frac{1}{K}(r + I_{zt}(1, \mu_{min}, \mu_{maj})) \qquad (7)$$

Where,

$$r = \sqrt{\frac{\sum_{c=1}^{K} N_c(\mu_c - \bar{\mu})^2}{\sum_{c=1}^{K}\sum_{i=1}^{N_c}(P_{ci} - \bar{\mu})^2}} \qquad \mu_c = \frac{\sum_{i=1}^{N_c} P_{ci}}{N_c} \qquad \mu_c = \frac{\sum_{i=1}^{N_c} P_{ci}}{N_c}$$

r computes the correlation ratio, where, µc represents the mean of classifier outputs for class c only, and µ□ represents the mean of µc for both minority and majority classes. As r only measures the separability of output values, indicator function Izt from eq (7) enforces the zero class threshold. In this case, Izt returns 1 if majority and minority class predictions are negative and non-negative, respectively, and 0 if otherwise.

2.3.4 Dist

It treats the genetic program outputs from the examples from the two classes as two independent distributions and measures the distance between these class distributions as the level of class separability. It computes the point equidistant from the means of two distributions, measured in terms of standard deviations away from the mean.

$$Dist = \frac{|\mu_{min} - \mu_{maj}|}{\sigma_{min} + \sigma_{maj}} \times I_{zt}(2, \mu_{min}, \mu_{maj}) \qquad (8)$$

Where, $\sigma_c = \sqrt{\frac{1}{N_c}\sum_{i=1}^{N_c}(P_{ci} - \mu_c)^2}$ $\mu_c = \frac{\sum_{i=1}^{N_c} P_{ci}}{N_c}$

μc and σc correspond to the mean and standard deviation of the class distribution c, where c is either the minority or majority class. It also uses the indicator function Izt to enforce the zero class threshold, here, the distance value for a solution is doubled if Izt returns 1.

3. PERFORMANCE ANALYSIS

This system is worked with two methods NB and SVM on two datasets pedestrian and Yeast. So in this project four types of systems are implemented. For this system two attributes accuracy and runtime are considered for comparison. Accuracy is calculated in %. It is plotted against generation, generation on X-axis and accuracy on Y-axis. Run time gives total time required to calculate accuracy of various fitness functions. It is measured in milliseconds. Experimental and comparative result of two systems are shown below.

Case I) Dataset: Pedestrian with method NB

i) Accuracy rate:

Here comparison of all old and new fitness functions with method NB for Pedestrian dataset is done. Comparative result is shown below in Table II. Graphical representation for Table II is shown below in Fig.1.

Table II: Comparison of Accuracy for Pedestrian with NB

Gen.	NB_Acc	NB_Ave	NB_Auc$_F$	NB_Auce	NB_Wmw	NB_Amse	NB_Incr	NB_Corr	NB_Dist
10	39.294117	78.117647	97.529411	91.235294	94.294117	75.147058	72.088235	81.661764	84.764705
20	40.470588	78.117647	97.735294	91.470588	94.602941	75.147058	72.088235	82.073529	84.764705
30	40.470588	78.352941	97.735294	91.470588	94.602941	75.147058	72.088235	82.073529	85.117647
40	40.764705	78.588235	97.735294	91.470588	94.602941	75.279411	72.088235	82.073529	85.117647
50	40.764705	78.588235	97.735294	91.470588	94.602941	75.279411	72.382352	82.073529	85.117647

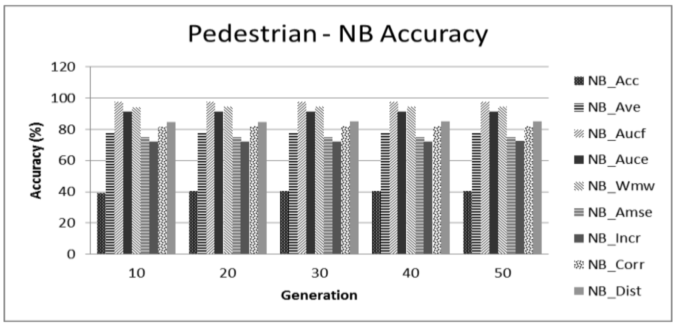

Figure 1: Graph of Comparison for Pedestrian with NB for Old-New FF

In comparison of old and new fitness functions Auc_F is the best fitness function as its accuracy is more than 97%. Second highest is *Wmw* and then *Auce* which is having 94% and 91% accuracy resp. In new fitness functions *Dist* is better fitness function as its accuracy is more than 85%. Lowest accuracy function is *Acc* which is having very poor accuracy *i.e.* 40%.

ii) Runtime

It is calculated in milliseconds. Table III shows time required to calculate accuracy of various fitness functions for Pedestrian dataset with NB and its graph shows in Fig 2.

Table III: Comparison of Runtime for Pedestrian with NB

Gen	**NB_Acc**	**NB_Ave**	**NB_Auc$_F$**	**NB_Auce**	**NB_Wmw**	**NB_Amse**	**NB_Incr**	**NB_Corr**	**NB_Dist**
10	11606	11684	11778	11856	11965	12043	12152	12230	12340
20	20623	20701	20795	20873	20951	21029	21107	21185	21263
30	35786	35974	36145	36317	36473	36644	36847	37066	37253
40	51199	51402	51589	51761	51948	52167	52369	52588	52791
50	66550	66659	66753	66862	66987	67080	67189	67283	67392

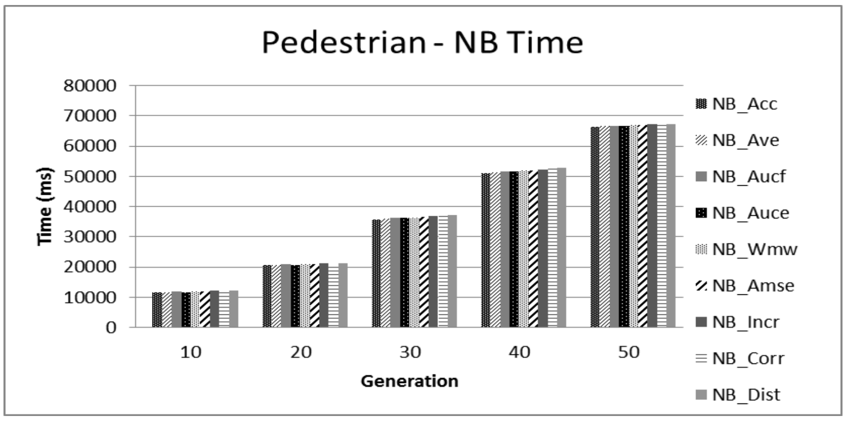

Figure 2: Graph of Runtime for Pedestrian with NB for Old-New FF

In comparison with old and new fitness functions, new fitness function *Dist* takes little bit more time than all other fitness functions. As generations are increasing runtime also increases.

Case IV) Dataset : Yeast with SVM

i) Accuracy rate:

Here comparison of all old and new fitness functions with method SVM for Yeast dataset is done. Comparative result is shown below in Table IV. In comparison of old and new fitness functions Auc_F is the best fitness function as its accuracy is more than 98%. Graphical representation for Table IV is shown below in Fig. 3.

Table IV: Comparison of Accuracy for Yeast with SVM

Gen.	SVM_Acc	SVM_Ave	SVM_Auc$_F$	SVM_Auce	SVM_Wmw	SVM_Amse	SVM_Incr	SVM_Corr	SVM_Dist
10	52.52941	82	98.61765	93.11765	95.83824	79.77941	76.5	84.95588	87.5
20	52.52941	82.23529412	98.61765	93.17647	95.88235	79.77941	77.08824	84.95588	87.76471
30	52.52941	82.58823529	98.64706	93.29412	95.97059	79.91176	77.23529	85.26471	87.94118
40	52.52941	82.58823529	98.64706	93.29412	95.97059	79.91176	77.23529	85.26471	87.94118
50	52.52941	82.58823529	98.64706	93.29412	95.97059	79.91176	77.23529	85.26471	87.94118

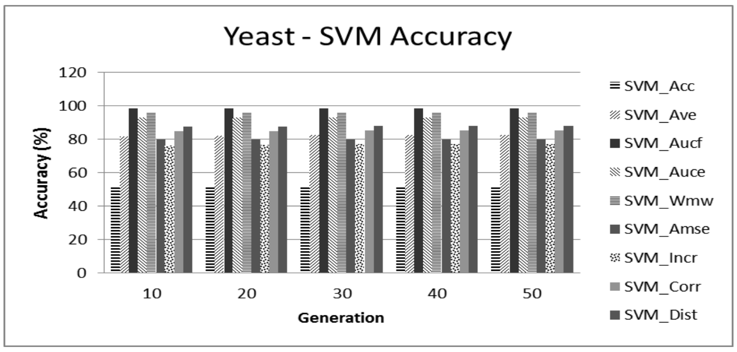

Figure 3: Graph of Comparison for Yeast with SVM for Old-New FF

In comparison of old and new fitness functions Auc_F is the best fitness function as its accuracy is up to 99%. Second highest is *Wmw* and then *Auce* which is having 96% and 94% accuracy resp. In new fitness functions *Dist* is better fitness function as its accuracy is more than 87%.

ii) Runtime

It is calculated in milliseconds. Table V shows time required to calculate accuracy of various fitness functions for Yeast dataset with SVM and its graph shows in Fig 4.

Table V: Comparison of Runtime for Yeast with SVM

Gen	SVM_Acc	SVM_Ave	SVM_Auc$_F$	SVM_Auce	SVM_Wmw	SVM_Amse	SVM_Incr	SVM_Corr	SVM_Dist
10	88483	89045	89637	90105	90605	91104	91556	92009	92492
20	129074	129480	129854	130213	130619	131009	131445	131867	132303
30	178168	178651	179119	179587	180086	180570	181054	181490	181912
40	234359	235045	235747	236402	237104	237853	238540	239273	240053
50	296353	296977	297711	298381	299505	300222	301080	301829	302609

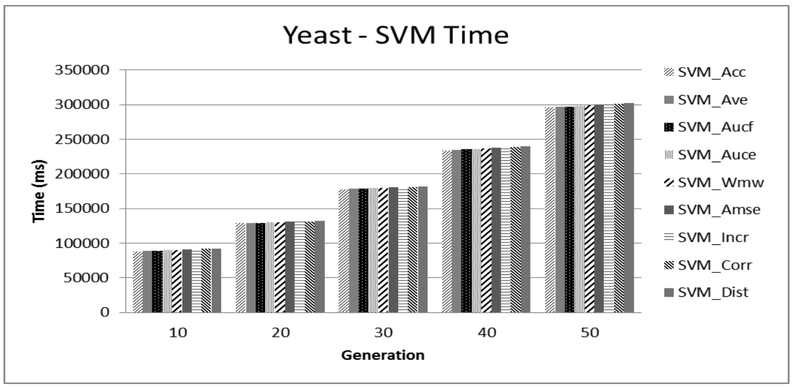

Figure 4: Graph of Runtime for Yeast with SVM for Old-New FF

In comparison with old and new fitness functions, new fitness function *Dist* takes more time than all other fitness functions. As generations are increasing runtime also increases.

4. CONCLUSION

This system proposed GP approach to binary classification with Imbalanced data focusing on cost adjustment within The fitness function in GP measures the overall performance of a solution by comparing the solutions predicted class label to the target or actual class label for all input instances. This system studies with NB and SVM with two datasets pedestrian and yeast. With SVM accuracy is better than NB. With SVM accuracy is 99% while with NB accuracy is 97%, but SVM takes five times more time than NB. Fitness function use in financial modeling such as insurance approval or bankruptcy prediction. Fitness function actively bias the distribution of exemplars over which learning is conducted. Implemented work can be extended with multiobjective GP for bagging algorithm.

REFERENCES

[1] Urvesh Bhowan, Mark Johnston and M. Zhang, “Developing new fitness functions in Genetic Programming for Classification with unbalanced data”,IEEE Trans On Sys, man and cybernetics—Part B: Vol. 42, No. 2, April2012.
[2] J. Eggermont, J.N.Kok, and W.A.Kosters, “Genetic Programming for data classification:Partitioning the search space”, in Proc. ACM SAC, 2004, pp.1001-1005.
[3] Yuchun Tang, Yan-Qing Zhang, Nitesh V. Chawla, and Sven Krasser, “SVMs modeling for highly imbalanced classification”,IEEE transactions on systems, man and cybernetics—part b: cybernetics, vol. 39, no. 1, February2009.
[4] 4. G. Patterson and M. Zhang, “Fitness functions in genetic programming for classification with unbalanced data”, in Proc. 20th Australian Joint Conf.Artif.Intell.,vol. 4830, LNCS, 2007.
[5] A.Asuncion and D. Newman, UCI machine learning repository, Irvine, CA.[Online]. Available: http://archive.ics.uci.edu/ml/MLRepository.html.
[6] P. G. Espejo, S. Ventura and F. Herrera, “A Survey on the application of genetic programming to classification”, IEEE Trans On Sys, Man, And Cybernetics—Part C: Vol. 40, No. 2, March 2010.
[7] S. Munder, D. M. Gavrila, “An experimental study on pedestrian classification using local features”, IEEE Trans on pattern analysis and machine intelligence, vol 28, no.11, Nov 2006.
[8] Paisitkriangkrai, C. Shen and J. Zhang, “An Experimental Study on Pedestrian Classification using Local Features”.
[9] VaishaliGanganwar, “An overview of classification algorithms for imbalanced Datasets”, IJETAE ISSN 2250-2459, Vol 2, Issue 4, April 2012.
[10] T. MaruthiPadmaja , NarendraDhulipalla , Raju S. Bapi , P.Radha Krishna,Unbalanced data classification using extreme outlier elimination and sampling techniques for fraud detection” ,15th International Conference on Advanced Computing and Communications.