K-Means Cluster Based Undersampling Ensemble for Imbalanced Data Classification

S. Santha Subbulaxmi, G. Arumugam

***Abstract:** Imbalanced data classification is a critical and challenging problem in both data mining and machine learning. Imbalanced data classification problems present in many application areas like rare medical diagnosis, risk management, fault-detection, etc. The traditional classification algorithms yield poor results in imbalanced classification problems.*

In this paper, K-Means cluster based undersampling ensemble algorithm is proposed to solve the imbalanced data classification problem. The proposed method combines K-Means cluster based undersampling and boosting method. The experimental results show that the proposed algorithm outperforms the other sampling ensemble algorithms of previous studies.

***Keywords:** imbalanced data, classification, undersampling, ensemble; k-means clustering*

I. INTRODUCTION

Imbalanced data classification is a critical and challenging problem in both data mining and machine learning. It is present in many application areas like rare medical diagnosis, risk management, fault-detection, etc. Some of the imbalanced data classification applications are, identifying infected people with a rare disease among the healthy people, detection of oil spills in satellite radar images, detection of fraudulent calls, detection of fraudulent credit cards transactions and detection of intrusion detection, etc.

The Imbalanced data have a skewed data distribution where the classes contain instances in different ratios. The imbalance ratio between the classes is represented with the class imbalance degree. The class which contains lesser instances is called as minority class and the class which contains more instances is called majority class.

When the traditional classification algorithms are applied over the imbalanced datasets, they assume that the dataset is normally distributed and tend to achieve better accuracy by being biased towards the majority class instances. In 2003, [1] Weiss GM conducted a study to explore the relationship between the classification performance of decision tree and the class distribution of a training data set. The study results proved that a relatively balance distribution usually attains better result. It implied the importance of handling the imbalance in the training data distribution. Another most common issue noticed in the traditional classification algorithms is, that they treat the misclassification errors in both majority and minority class as equal and ignore the misclassification cost of minority class instances. The imbalance degree and the complexity factors like small sample size, class overlap, and within class concepts have a predominant effect in the performance of classifier.

Revised Manuscript Received on February 22, 2020.

* Correspondence Author
Ms. S. Santha Subbulaxmi,* Research Scholar, Madurai Kamaraj University, Madurai, Tamil Nadu, India, Email: santhamd3@gmail.com
Dr. G. Arumugam, Professor & Head, Department of (Retd.), Computer Science, Madurai Kamaraj University, Madurai, Tamil Nadu, India.
Email: gurusamyarumugam@gmail.com

Small sample size

The sample size plays a crucial role in imbalanced data classification. If the sample size is limited, the imbalanced data classification performance deteriorates. [2] In 2002, a study was conducted to check the sample size and the classifier performance. The classifiers used in the study were decision tree, neural networks, and support vector machines. The experiment results had proven that as the size of the training set increases, the error rate caused by the imbalance class distribution decreases.

Overlap

In imbalanced dataset, the real challenge is to separate and identify the minority class from the majority class. If the classes are overlapping at different levels in some feature space, the classification becomes difficult. [3] Prati conducted several experiments on imbalanced datasets with class overlap and concluded that when the classes are sharing highly overlapped regions, the number of minority class instances being correctly classified will be decreasing.

Within-class concepts

The real word imbalanced data sets used to contain various sub-concepts in a single class. These sub-concepts do not always contain the same number of instances which result in within-class imbalance. [4] Japkowicz conducted a set of experiments to check the with-in class concepts problem and concluded that the presence of sub-concepts increased the complexity.

Several approaches were proposed to address the issues in imbalance classification. We propose an effective ensemble method to undersample the majority class and solve the imbalanced data classification problem. In summary the contributions of this paper are as follows:

- An effective undersampling ensemble method is proposed to address the imbalance in binary class imbalanced data sets.

- Experimental analysis is provided to prove the effectiveness of the proposed algorithm with existing undersampling methods.

The remaining part of this paper is organized as follows. In Section 2, we have presented important studies in imbalanced data classification problem and studies related to our proposed method. The proposed method is presented in Section 3. The experimental results are discussed in Section 4. Finally, in Section 5 we make our concluding remarks.

II. IMPORTANT STUDIES AND RELATED WORK

The algorithms, decision trees, support vector machines, neural networks, Bayesian network, nearest neighbour are widely used as base classifiers in the imbalance classification problems. To improve the performance of classifiers, several approaches were proposed and experimented. These approaches can grouped into data level approach, algorithm level approach and cost sensitive level. The data level approach, attempts to solve the problem through data reduction or by resampling the data space or by applying feature engineering concepts, etc. The algorithm level approach, aims to solve the problem through modifying the learning process of the classifiers. The cost sensitive level approaches uses cost matrix and instance weighting to reduce the misclassification cost.

Data reduction algorithms are used to reduce the majority class instances and refine the imbalance distribution to improve the performance of the classifier. The algorithms, [5] Tomek links, [6] Condensed nearest neighbour (CNN), [7] Reduced nearest neighbour (RNN), [8] Selective nearest neighbour (SNN), [9] Edited Nearest Neighbour (ENN) , [10] Neighbourhood cleaning rule, [11] Repeated ENN (RENN), [12] are the popular data reduction algorithms.

The sampling algorithms are instance level algorithms. It can be categorized into oversampling and undersampling, Hybrid Sampling is the combination of both oversampling and undersampling.

The oversampling techniques replicate the existing instances or add synthetic instances to balance the distribution. The random oversampling technique replicates the instances of minority classes randomly until the data distribution is balanced. [13] Synthetic Minority Oversampling Technique (Smote) generates new synthetic instances in minority class by interpolating the k nearest neighbours and the synthetic instances are generated until the data distribution is balanced. [14] SMOTEBoost algorithm generates new minority instances and applies boosting to focus more on the difficult minority class instances. [15] SMOTEBagging randomly oversamples the minority class instances and creates each bag significantly different to achieve better performance. Oversampling techniques are prone to overfitting and require longer time for training.

Undersampling algorithms reduce the number of the instances in majority class randomly or chooses a representative subset R from the population training set S to balance the distribution. [16] In random under sampling, instances of majority class are randomly chosen to derive R. If the reduction or selection is done by using statistical knowledge then it is called as informed undersampling.

Clustering based undersampling handles both the inter and intra class imbalance well. When the subsets inside a class is misappropriate, it results in Intra-class imbalance distribution. [17] Yen SJ, proposed a cluster based undersampling algorithm - SBC. The SBC method first clusters all the training samples into some clusters. Based on the imbalance ratio exists in the cluster, it selects appropriate majority instances from the clusters by using five different approaches. They are clustering with NearMisss-1, NearMisss-2, NearMisss-3, sampling based on clustering with Most Distance, sampling based on clustering with most far to choose the majority instances. CBO is a popular cluster based undersampling algorithm [18]. It applies clustering techniques on the instances in majority class and instances in minority class separately. Two sets of clusters are generated for each class. CBO identifies the largest cluster in the majority class. It applies random oversampling to all majority class clusters except the largest majority cluster. It oversamples those clusters until they have the same number of instances as the largest majority cluster. CBO applies random oversampling to all the clusters in the minority class until the total number of instances in the minority class equals to the total number of instances in the majority class. [19] ClusterOSS is an enhancement of One Sided Selection algorithm. Using a clustering algorithm, it first clusters the instances of majority class. In each cluster, instances near to the cluster centre are used to initiate the sampling procedure. Tomek links is used to clean the borderline and noisy majority instances. The ensemble learning techniques are often used in undersampling for better performance. [20] Easyensemble and balancecascade are the two popular ensemble-based-undersampling algorithms. [21] UnderBagging randomly chooses majority instances and constructs k classifiers and predicts the class which gets the most votes. [22] RUSBoost chooses majority instances randomly in each iteration and assign weights to the instances by normalizing the weights with their total sum weights.

The proposed cluster based undersampling algorithm aims to solve the imbalanced data classification problem by undersampling the majority class instances through clustering principle. It also avail the benefit of randomness, iterative and ensemble computing power. Thus, it differs from the existing cluster based undersampling algorithms.

III. PROPOSED METHOD

In this section, we have presented our method for solving the binary class imbalanced data classification problem. It is applicable to only the numeric datasets. Our method derives a representative majority class dataset and balances the data distribution. The representative majority class data set is derived by grouping and subgrouping the majority class instances and then choosing the instances randomly in each sub group. The process of selecting the majority class instances from each group is unique and it differs from the existing methods. For grouping the instances, K-Means clustering is used and K clusters are constructed. For each cluster, the euclidean distance between the cluster instances and cluster centroid is calculated.

The euclidean distance distribution of each cluster is then sub grouped by using the measure “Deciles”. 10 quantiles are called as deciles. The quantiles are the cut points in a distribution which relate to the rank order of values in that distribution. The quantiles have specific names too. For example, the 100 quantile is called as percentiles.

The representative training dataset is constructed by combining the majority class instances chosen from each subgroup in the clusters and the minority class instances. To avail the benefit of randomness in the choice of majority class instances, the representative dataset is constructed iteratively for N times in the experiment.

The Support Vector Machine (SVM) algorithm is used to train the weak learners. The ensemble technique, boosting is used to improve the performance of the weak learner. This algorithm adapts the boosting technique used in RUSBoost[22]. The boosting technique used in RUSBoost[22] is based on SMOTEBoost[14] which is in turn based on AdaBoost.M2.

The proposed method is shown in Fig 1 and described as below:

1. **Partition the data into groups and subgroups**

The training data set is partitioned into majority and minority class data sets. The imbalance ratio, IRsize is calculated by dividing the number of minority class instances with the number of majority class instances. The training data set is scaled with z-score. The instances in training set are grouped by applying the K-Means clustering algorithm. For this, the optimal number of K (number of clusters) is derived based on the silhouette method. The training dataset is then grouped as K clusters using K-Means algorithm and Traincluster is derived. The Euclidean distance of each instance with its cluster centroid is calculated. The Traincluster is added with two new fields euclidean distance list and subgroupno. The field subgroupno is set as zero. Then the decile subgrouping method partitions each cluster (group) into subgroups. The decile subgrouping method is applied based on the calculated Euclidean distance of each instance with its centroid.

The decile subgrouping method contains the following steps:

a. The euclidean distances of the instances is converted into continuous intervals with equal probabilities

b. The cut points for the subgroup number =10 (decile) of the euclidean distance distribution are calculated.

c. The instances which fall within the range of cut points are grouped and hence 10 subgroups are created in each cluster and the subgroupno is updated in Traincluster accordingly.

2. Create a representative dataset by subgrouping and build the ensemble

The representative data sets creation process starts by assigning weights to the instances in the training set. A weight field is added in the training set and the initial weights are merely the inverse of the training set size. The Traincluster is also updated by the same weight of training set. The number of majority instances and minority instances in each cluster and subgroups is calculated and Local_IRsize is derived. The Local_IRsize is calculated as no. of minority instances in the subgroup/no. of majority instances in the subgroup. The no. of majority instances to be chosen from the subgroup is calculated as no. of majority instances in the subgroup * max(IRsize * Local_IRsize) and then that number of majority class instances are randomly chosen from the sub group. The representative majority set is created by matching instances in Training dataset for the randomly selected majority instances. The representative majority set is combined with minority data set and a representative training set is formed. The weight in the representative majority dataset and training dataset is updated as Null. A weak model is built on the representative training set by using Support Vector Machine (SVM). The weak model is evaluated with the original training set and new weights are computed based on the pseudo loss. Then the new weights are normalized and reassigned to the instances in training set as well as in the Traincluster. The boosting iteration is repeated for the defined Iteration_number times and weak learners are created. At the end, an ensemble is built.

IV. EXPERIMENTAL RESULTS

In this section we have presented and discussed about the experimental results achieved by our proposed method. The proposed Undersampling ensemble algorithm is developed using R 3.2. K-Means algorithm is implemented by using the R package “Cluster” and Support vector machine is implemented by using the R package “e1071”. The experiments are conducted on 10 data sets from Keel data repository [23]. The dataset name, number of features, total number of instances and imbalance ratio are summarized in Table-I.

Table- I: Details of the datasets

S.No.	Dataset	#Features	Total #Instances	Imbalance Ratio
1	wisconsin	9	683	1.86
2	glass-0-1-2-3_vs_4-5-6	9	214	3.2
3	page-blocks0	10	5472	8.79
4	yeast-2_vs_4	8	514	9.08
5	yeast-1_vs_7	7	459	14.3
6	yeast-1-2-8-9_vs_7	8	947	30.57
7	yeast5	8	1484	32.73
8	ecoli-0-6-7_vs_3-5	7	222	9.09
9	yeast-0-2-5-7-9_vs_3-6-8	8	1004	9.14
10	ecoli-0-1_vs_2-3-5	7	244	9.17

The number of features of the datasets range from 7 to 10. The number of instances in the data sets ranges from 214 and 5472. The imbalance ratio of the data sets ranges from 1.86 and 32.73. The datasets are partitioned into training (80%) and test datasets (20%). The test dataset is scaled with z-score based on the training dataset mean and standard deviation. The scaled test dataset is predicted with the proposed method and results are estimated.

From the results, the probability of being a minority instance is estimated and averaged. If the averaged probability of the test instance is larger than 0.5, then it is predicted as minority and in the other case it is predicted as majority.

In the experiment, the prediction capacity of the proposed undersampling algorithm is assessed for the 10 datasets with the help of the performance metric "AUC" (Area Under Curve) and G-Mean. The AUC shows up the discrimination performance of the model under different thresholds. The ROC and AUC are measured using the "PROC" package.

The proposed algorithm is compared with the other popular ensemble algorithms Underbagging, RUSBoost, Smotebagging and Smoteboost. These popular sampling ensemble algorithms are implemented using the "ebmc" package. Table-II shows up the comparison results of AUC value of the algorithms.

Table- II: Performance Evaluation in terms to AUC with Proposed Undersampling algorithm and the Popular Ensembling Algorithms

S.No.	Dataset	Proposed	Under Bagging	RUSBoost	SMOTE Bagging
1	wisconsin	0.991667	0.978721	0.981481	0.978721
2	glass-0-1-2-3_vs_4-5-6	1	0.97619	0.97619	0.97619
3	page-blocks0	0.945515	0.832656	0.851459	0.828045
4	yeast-2_vs_4	0.973958	0.901572	0.901572	0.945409
5	yeast-1_vs_7	0.977528	0.528571	0.569444	0.569444
6	yeast-1-2-8-9_vs_7	0.983957	0.559878	0.636034	0.534985
7	yeast5	0.988014	0.871466	0.925063	0.849395
8	ecoli-0-6-7_vs_3-5	1	0.875	0.97561	0.875
9	yeast-0-2-5-7-9_vs_3-6-8	0.931667	0.877619	0.824612	0.858095
10	ecoli-0-1_vs_2-3-5	0.988636	0.821705	0.977778	0.821705

The AUC comparison shows up that the proposed algorithm outperforms with UnderBagging, RUSBoost, SMOTEBagging, SMOTEBoost.

Table-III shows up the comparison results of GMean value of the algorithms

Table- III: Performance Evaluation in terms to GMEAN with Proposed Undersampling algorithm and the Popular Ensembling Algorithms

S.No.	Dataset	Proposed	UnderBagging	RUSBoost	SMOTE Bagging
1	wisconsin	0.993569	0.976584	0.974245	0.976584
2	glass-0-1-2-3_vs_4-5-6	1	0.942809	0.942809	0.942809
3	page-blocks0	0.834395	0.931437	0.940051	0.930333
4	yeast-2_vs_4	0.816497	0.846886	0.846886	0.889499
5	yeast-1_vs_7	0.447214	0.561511	0.702935	0.702935
6	yeast-1-2-8-9_vs_7	0.5	0.484568	0.600414	0.47594
7	yeast5	0.707107	0.9193	0.922566	0.917663
8	ecoli-0-6-7_vs_3-5	1	0.987096	0.57735	0.987096
9	yeast-0-2-5-7-9_vs_3-6-8	0.930233	0.940331	0.908968	0.917
10	ecoli-0-1_vs_2-3-5	0.816497	0.806947	0.57735	0.806947

The GMean comparison shows up that the proposed algorithm outperforms with 4 times with all the four algorithms. It outperforms 6 times with SMOTEBagging. It outperforms 5 times with UnderBagging, RUSBoost and SMOTEBoost.

The results of the proposed method have performance advantages compared to the popular ensemble algorithms Underbagging, RUSBoost, Smotebagging and Smoteboost. The proposed method outperforms in most of the datasets with imbalance ratio of 9 and above. The majority class instances chosen for the sample data set are based on the local imbalance ratio exists in the subgroups. So the sample data set best represents the training data distribution and provide better results than the other methods.

V. CONCLUSION

The imbalanced data classification problems exist widely in the real world applications. It is a predominant problem in machine learning and data mining. The high dimensions, small sample size, class overlap and sub concepts within class are the available challenges in imbalanced data classification. Several competent methods have been reported in the literature to address these challenges. The proposed method is a competent method to solve the class imbalance problem. It gives importance to local imbalance ratio and balances the distribution. The sample data sets generated best represents the data distribution. Hence it achieves good performance results while comparing it to the existing popular algorithms.

Algorithm: K-Means Cluster based Undersampling Ensemble

```
--------------------------------------------------------------------------------------------------------------
Input: Training dataset of an Imbalanced Binary Class dataset $D_t$, Iteration_number (By Default 20)
Output: An Undersampling Ensemble Classifier
1. Examine the training data set and scale the Training Dataset with z-score
2. Partition Training Dataset into majority data set (Ma) and minority data set (Mi)
3. Let Mi_Size=No. of instances in Mi
4. Let Ma_Size=No. of instances in Ma
5. Calculate IRsize = Mi_Size / Ma_Size
6. Calculate K=No. of optimal K-means clusters of training dataset by using Silhouette method
7. Let Traincluster=Apply K-means Clustering Algorithm to create K clusters of training dataset
8. Euclidean_distance_list[]=Calculate the Euclidean distance of each instance in Traincluster with its cluster centroid
9. Add two new fields in Traincluster as edistance=euclidean_distance_list[], subgroupno=list of zeros
10. For I=1 to K
        a. Let Csize=Number of instances in Traincluster for clusterno=I
        b. If (Csize<10) then
                i. Update subgroupno with 1 in Traincluster for clusterno=I
        c. Else if (Csize>10) then
                i. Let cut_points[] = calculate decile cutpoints of the euclidean distance of the instances in Traincluster for
                   clusterno=I
                ii. For J=1 to 10
                        1. Let Subgroup= Get Traincluster instances in the distance between cutpoints[J] and
                           cutpoints[J+1] for clusterno=I
                        2. Update subgroupno with J in Traincluster for matching instances in Subgroup
                iii. Next
        d. End if
11. Next
12. Let weights of instances in training set ($D_t$)= 1/(number of instances in the training set)
13. Set T= Iteration_number
14. For t=1 to T
        a. Representative majority set = KCluster_Decile_LocalIR_Undersample (Training data set, Traincluster, K,
           IRsize),
        b. Let Representative training set = Combination of Representative majority set and minority set
        c. Update weights in the Representative training set as Null
        d. Update weights in the training data set as Null
        e. Train the representative training set with SVM and Build weak Model
        f. Evaluate the weak model with representative training set and Update new weights in the Data distribution
                i. Let the hypothesis $h_t : X \times Y \rightarrow [majorityclass, minorityclass]$
                ii. Compute pseudo loss as $\epsilon_t = \sum_{(i,y):y_i \neq y} D_t(i)\left(1 - h_t(x_i, y_i) + h_t(x_i, y)\right)$
                iii. Set the weight update parameter $\alpha_t = \epsilon_t / (1 - \epsilon_t)$
                iv. Update $D_t$: $D_{t+1}(i) = D_t(i)\ \alpha_t^{\frac{1}{2}(1+h_t(x_i,y_i)-h_t(x_i,y:y\neq y_i))}$
                v. Normalize and update weights in Training dataset $D_{t+1}(i) = \frac{D_{t+1}(i)}{\sum_i D_{t+1}(i)}$
                vi. Update the same weights in Traincluster
15. Next t
16. Build the ensemble   $\arg\max_{y \in Y} \sum_{t=1}^{T} \left(\log \frac{1}{\alpha_t}\right) . h_t(x, y)$
```

Fig. 1. ***K-Means Cluster based Undersampling Ensemble***

Algorithm: KCluster_Decile_LocalIR_Undersample

```
--------------------------------------------------------------------------------------------------------------
Input: Training dataset, Traincluster, K, IRsize
Output: A representative majority class data set
1. Let Representative majority class data set = empty data set
2. Let Random_subgroup instances=empty dataset
3. For I=1 to K
        a. Csize=No. of instances in Traincluster for the clusterno=I
        b. If Csize< 10 then
                I. Let subgroup= Instances in Traincluster for clusterno=I
```

```
            II.   Let subgroup_size= Csize
            III.  Let Mi_size1=number of minority instances in subgroup
            IV.   Let Ma_size1=number of majority instances in subgroup
            V.    Local_IRsize= Mi_size1/ Ma_size1
            VI.   Sel_IRsize=max(IRsize, Local_IRsize)
            VII.  SI=Sel_IRsize * Ma_size1
            VIII. Append Random_subgroup instances with randomly chosen SI number of majority instances from
                  subgroup based on the weights assigned to the instances in Traincluster
      c. Else
            I.    For J=1 to 10
                     i.    Let subgroup= Instances in Traincluster for clusterno=I and subgroupno=J
                     ii.   Let subgroup_size= number of instances in subgroup
                     iii.  Let Mi_size1=number of minority instances in subgroup
                     iv.   Let Ma_size1=number of majority instances in subgroup
                     v.    Local_IRsize= Mi_size1/ Ma_size1
                     vi.   Sel_IRsize=max(IRsize, Local_IRsize)
                     vii.  SI=Sel_IRsize * Ma_size1
                     viii. Append Random_subgroup instances with randomly chosen SI number of majority instances from
                           subgroup based on the weights assigned to the instances in Traincluster
            II.   Next J
      d. End if
4. Next I
5. Let Representative majority class data set =Majority instances in Training dataset for matching the Random_subgroup
   instances
6. Return Representative majority class data set
```

Fig. 2.*KCluster_Decile_LocalIR_Undersample Method*

REFERENCES

1. G. M. Weiss and F. Provost, "Learning when training data are costly: The effect of class distribution on tree induction," Journal of artificial intelligence research, vol. 19, pp. 315–354, 2003.
2. N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," Intelligent data analysis, vol. 6, no. 5, pp. 429–449, 2002.
3. R. C. Prati, G. E. Batista, and M. C. Monard, "Class imbalances versus class overlapping: an analysis of a learning system behavior," in Mexican international conference on artificial intelligence, 2004, pp. 312–321.
4. N. Japkowicz, "Concept-learning in the presence of between-class and within-class imbalances," in Conference of the Canadian society for computational studies of intelligence, 2001, pp. 67–77.
5. I. Tomek, "Two modifications of CNN," IEEE Trans. Systems, Man and Cybernetics, vol. 6, pp. 769–772, 1976.
6. T. Cover and P. Hart, "Nearest neighbor pattern classification," IEEE transactions on information theory, vol. 13, no. 1, pp. 21–27, 1967.
7. G. Gates, "The reduced nearest neighbor rule (Corresp.)," IEEE transactions on information theory, vol. 18, no. 3, pp. 431–433, 1972.
8. G. Ritter, H. Woodruff, S. Lowry, and T. Isenhour, "An algorithm for a selective nearest neighbor decision rule (Corresp.)," IEEE Transactions on Information Theory, vol. 21, no. 6, pp. 665–669, 1975.
9. D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," IEEE Transactions on Systems, Man, and Cybernetics, no. 3, pp. 408–421, 1972.
10. J. Laurikkala, "Improving identification of difficult small classes by balancing class distribution," in Conference on Artificial Intelligence in Medicine in Europe, 2001, pp. 63–66.
11. I. Tomek, "An experiment with the edited nearest-neighbor rule," IEEE Transactions on systems, Man, and Cybernetics, vol. 6, no. 6, pp. 448–452, 1976.
12. I. Mani and I. Zhang, "kNN approach to unbalanced data distributions: a case study involving information extraction," in Proceedings of workshop on learning from imbalanced datasets, 2003, vol. 126.
13. N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," Journal of artificial intelligence research, vol. 16, pp. 321–357, 2002.
14. N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: Improving prediction of the minority class in boosting," in European conference on principles of data mining and knowledge discovery, 2003, pp. 107–119.
15. S. Wang and X. Yao, "Diversity analysis on imbalanced data sets by using ensemble models," in 2009 IEEE Symposium on Computational Intelligence and Data Mining, 2009, pp. 324–331.
16. N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," Intelligent data analysis, vol. 6, no. 5, pp. 429–449, 2002.
17. S.-J. Yen and Y.-S. Lee, "Cluster-based under-sampling approaches for imbalanced data distributions," Expert Systems with Applications, vol. 36, no. 3, pp. 5718–5727, 2009.
18. T. Jo and N. Japkowicz, "Class imbalances versus small disjuncts," ACM Sigkdd Explorations Newsletter, vol. 6, no. 1, pp. 40–49, 2004.
19. V. H. Barella, E. P. Costa, and A. Carvalho, "ClusterOSS: a new undersampling method for imbalanced learning," in Brazilian Conference on Intelligent Systems, 3rd; Encontro Nacional de Inteligência Artificial e Computacional, 11th. Universidade de São Paulo-USP, 2014.
20. X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 39, no. 2, pp. 539–550, 2008.
21. S. Wang and X. Yao, "Diversity analysis on imbalanced data sets by using ensemble models," in 2009 IEEE Symposium on Computational Intelligence and Data Mining, 2009, pp. 324–331.
22. C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 40, no. 1, pp. 185–197, 2009.
23. J. Alcalá-Fdez et al., "Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework.," Journal of Multiple-Valued Logic & Soft Computing, vol. 17, 2011.

AUTHORS PROFILE

Ms. S. Santha Subbulaxmi is a Research Scholar in Madurai Kamaraj University, Madurai, Tamil Nadu, India.E-mail: santhamd3@gmail.com

Dr. G.Arumugam is Professor & Head of the Department (Retd.), Computer Science Department, Madurai Kamaraj University, Madurai, Tamil Nadu, India.E-mail: gurusamyarumugam@gmail.com