International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-5, January 2020

K-Means Cluster Based Oversampling Algorithm for Imbalanced Data Classification

S. Santha Subbulaxmi, G. Arumugam

***Abstract*: Imbalanced data classification problems endeavor to find a dependent variable in a skewed data distribution. Imbalanced data classification problems present in many application areas like, medical disease diagnosis, risk management, fault-detection, etc. It is a challenging problem in the field of machine learning and data mining.**

In this paper, K-Means cluster based oversampling algorithm is proposed to solve the imbalanced data classification problem. The experimental results show that the proposed algorithm outperforms the existing oversampling algorithms of previous studies.

***Keywords* : imbalanced data, classification, oversampling, k-means clustering, synthetic instances**

I. INTRODUCTION

Imbalanced data classification problems aim to find the dependent variable in a skewed data distribution. The skewed data distribution results in class imbalance. The dataset may be of a binary class dataset or a multi class dataset. A binary class imbalanced dataset contains many more instances in one class (majority class) and lesser number of instances in another class (minority class). A multi-class imbalanced dataset contains many majority classes and a minority class. The traditional classification algorithms tend to be biased towards majority class by discriminating the minority class instances. They also treat misclassification cost as equal in both majority and minority classes. The data difficulty factors like high dimensions, small disjuncts, and small sample size lead the imbalanced data classification problem more complicated.

Imbalanced data classification problems exist in various fields like medical, industry, business, etc., Intrusion detection in networks, fault detection in Instruments, medical complication risk identification, financial risk identification are, few of the real world applications of imbalanced data classification. There exist several approaches to deal with the imbalanced data classification problems. They can be broadly categorized into data level approaches, algorithmic level approaches and cost sensitive learning level approaches. The data level approaches include data cleaning, sampling and feature engineering methods. The algorithmic level approaches used to alter the learning process of the learners. The cost sensitive learning level approaches used to modify algorithms to reduce misclassification cost or provide weights to instances or both.

Most of the data cleaning algorithms used to find and remove majority class instances which are in the neighborhood of minority class instances. [1] Tomek links, [2] Neighbourhood cleaning rule (NCL) are the popular data cleaning algorithms. The sampling algorithms can be categorized as undersampling algorithms and oversampling algorithms. The drawback of undersampling algorithms is, they may loss few of the important information in majority class instances by removing it. The drawback of oversampling algorithms is, they are prone to be overfitting and increases the computational cost.

In this paper, we propose an effective oversampling method to solve the imbalanced data classification problem. The remaining part of the paper is organized as follows. Section 2 describes few of the important studies in the oversampling approach. Section 3 details about the proposed method and discusses the results. Section 5 concludes with concluding remarks and future plan.

II. RELATED WORKS

The research literature related to oversampling methods are discussed in this section. The Random oversampling replicates the instances in minority class and balances the data distribution. [3] Synthetic Minority Oversampling Technique (SMOTE) generates new synthetic instances in a minority class instance by interpolating with its nearest neighbours and balances the data distribution. [4] Random-SMOTE generates synthetic instances randomly in minority class. [5] MSMOTE computes the distances of all the minority class instances and groups them as security instances, border instances and latent noise instances. For the security group instances, it selects a nearest neighbor instance randomly among the K neighbors and generates synthetic instances. For a border instance, it selects a nearest neighbor and generates synthetic instances. [6] The safe-level-SMOTE calculates the safe level ratio of each minority class instance. The safe level ratio of minority class instance is based on the ratio of minority class instances in its neighborhood. The instances which have highest safe level ratio are used to create synthetic instances. [7]

Manuscript published on January 30, 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 of the Department (Retd.), Department of Computer Science, Madurai Kamaraj University, Madurai, Tamil Nadu, India. gurusamyarumugam@gmail.com

Borderline SMOTE is the enhancement of SMOTE. It categorizes the minority instances as Noise and Danger based on its neighborhood. If the neighborhood of minority instances contains only majority instances, then the minority instance is called as noise. If the neighborhood of the minority instances contains equal number of instances from both majority and minority classes, then the minority instance is called as Danger.

The SMOTE algorithm is applied on each Danger instance and synthetic instances are created. [8] Hien M and etal developed another Borderline SMOTE algorithm which has a similar approach in categorizing the minority instances as Danger and Noise. In this algorithm, synthetic samples are created between Danger and its nearest neighbors. [9] Neater creates synthetic instances using SMOTE and then uses game theory to clean and balance the data distribution. [10] Jorge uses a randomized weighted distance scheme and generates synthetic instances in the neighborhood of each minority instances. [11] The ADASYN method creates synthetic instances in minority class based on the majority class instances in the data distribution. [12] The Radial based oversampling method uses radial basis functions to identify suitable areas in minority class and generate synthetic instances. [13] K-means SMOTE uses K-means clustering method to cluster the data and targets the critical areas. It assigns more weight to sparse minority clusters and generates synthetic instances. The [14] DBSMOTE method uses DBSCAN clustering algorithm and a density reachable graph to create synthetic instances. [15] A-SUWO method uses hierarchical clustering method and oversamples the minority instances which are close to the decision boundary.

The proposed method uses, K-Means clustering algorithm to identify the sub concepts in minority class and creates synthetic instances around the neighborhood of class within sub concepts in minority class.

III. PROPOSED METHOD

In this section, we have presented the proposed oversampling method. It is applicable to binary class numeric datasets. Our proposed method balances the distribution by creating synthetic instances around the neighborhood of class within sub concepts in minority class. Hence, our method is unique and differs from the existing methods.

The K-Means cluster algorithm is used to group the minority class instances. The euclidean distance of each minority class with its centroid is computed. The deciles of the euclidean distance distribution are computed and subgroups are created based on the computation. 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. Synthetic instances are created around the minority instances in the subgroup and its nearest neighbours. To find out the nearest neighbours KNN query is used. The Support Vector Machine (SVM) algorithm is used to train the model. The proposed method is shown in Fig 1 and described as below:

1. Partition the Training Dataset

The training dataset is scaled with z-score. The training dataset is partitioned into majority (Ma) and minority (Mi) class datasets. The Ma_size denotes the number of instances in majority class and Mi_size denotes the number of instances in minority class.

2. Identify the groups and subgroups in the minority class

To identify the class within sub concepts in the minority class, K-Means algorithm is used. Silhouette method is used to find out the optimal number of K. The minority class instances are grouped into K clusters by using K-Means algorithm and Minoritycluster is derived. In each cluster, the euclidean distance between the minority instances in that cluster with its cluster centroid is calculated. The Minoritycluster is added with two new fields: euclidean distance list and subgroupno. The subgroupno is set as zeros. The instances in minority cluster are further divided into subgroups based on the decile cut points of the euclidean distance distribution and subgroupno is updated accordingly. The matching instances in minority class are retrieved for each clusterno and subgroupno of the Minoritycluster and synthetic instances are created around the neighborhood of minority class instances in each subgroup.

3. Build synthetic datasets

The single instances in each cluster and subgroup cannot be used to create synthetic instances and the number of single instances is denoted as Si_Size. The other minority instances should generate a certain number of synthetic instances. The synthetic instance rate (SR) of each minority instance is calculated with the equation (1).

SR=(Ma_size-Mi_Size)/ (Mi_size-Si_Size) ----------(1)

Each minority class instance has to generate SR rate of synthetic instances. The neighbors of each minority instance (DK) are identified using a KNN query. If the neighbor count (DK_size) is less than the synthetic instance rate (SR), the minority instance interpolates with the first neighbor and SR synthetic instances are created. In the other case, the Local synthetic instance count (LSR) is calculated by dividing SR with the number of neighbors. Each minority instance will interpolate with its neighbor for LSR times and generate SR synthetic instances. For the interpolation, the difference between the feature values of a minority instance and the feature values of its minority neighbors are computed. The difference is multiplied with a random number and then added with the feature values of the specific minority instance. The generated synthetic instances from each subgroup are combined and the final synthetic minority dataset is derived.

4. Obtain a representative training set

A representative training dataset is obtained by combining the minority class dataset, majority class dataset and the final synthetic minority dataset.

5. Build the model.

A model is built based on the representative training dataset by using the Support Vector Machine algorithm.

Algorithm:K-Means Cluster based oversampling Method

Input: Training dataset of an Imbalanced Binary Class Dataset
Output: An Oversampling Classifier

1. Examine the training dataset and scale the Training Dataset with z-score
2. Partition Training Dataset into Majority class dataset (Ma) and Minority class dataset (Mi)
3. Let Mi_Size=No. of instances in Mi
4. Let Ma_Size=No. of instances in Ma
5. Calculate K=No. of optimal K-means clusters of Minority dataset by using Silhouette method
6. Let MinorityCluster=Apply K-means Clustering Algorithm to create K clusters of Minority dataset
7. Let Euclidean_distance_list[]=Calculate the Euclidean distance of each instance in MinorityCluster with its cluster centroid
8. Add two new fields in Minoritycluster as edistance=euclidean_distance_list[], subgroupno=list of zeros
9. Let Si_C=0
10. Let Si_SG=0
11. For I=1 to K
    a) Let Csize=Number of instances in Minoritycluster for clusterno=I
    b) If (Csize=1) then
        I. Let Si_C=Si_C+1
        II. Update subgroupno with 1 in Minoritycluster for clusterno=I
    c) Else if (Csize>10) then
        I. Let cut_points[] = calculate decile cutpoints of the euclidean distance of the instances in Minoritycluster for clusterno=I
        II. For J=1 to 10
            i. Let Subgroup= Get Minoritycluster instances in the distance between cutpoints[J] and cutpoints[J+1] for clusterno=I
            ii. Update subgroupno with J in Minoritycluster for matching instances in Subgroup
            iii. Let SG_size=Number of instances in Subgroup
            iv. If (SG_size=1) then Let Si_SG=Si_SG+1
        III. Next
    d) End if
12. Next
13. Let Si_Size= Si_C + Si_SG
14. Let SR= (Ma_Size-Mi_Size)/ (Mi_Size-Si_Size)
15. Let GS_dataset = empty
16. For I=1 to K
    a) Let Csize=Number of instances in Minoritycluster for clusterno=I
    b) If (Csize <10) then
        I. Let Dec_instances[]=Get matching instances from Minority dataset for the instances in Minoritycluster for clusterno=I
        II. Let Dec_size= Csize
        III. If (Dec_size>1) then
            i. Let Syn_instances = Generate_Syntheticinstances(Dec_instances[], SR, Dec_size)
            ii. Append Syn_instances in GS_dataset
    c) Else
        I. For J=1 to 10
            i. Let Dec_instances[]=Get matching instances from Minority dataset for the instances in Minoritycluster for clusterno=I and subgroupno=J
            ii. Let Dec_size = Number of instances in Dec_instances[]
            iii. If (Dec_size>1) then
                1. Let Syn_instances= Generate_Syntheticinstances (Dec_instances[], SR, Dec_size)
                2. Append Syn_instances in GS_dataset
        II. Next
    d) End if
17. Next
18. Let synthetic minority dataset = GS_dataset
19. Let representative training dataset = combine minority dataset, majority dataset, synthetic minority dataset
20. Train the representative training dataset with SVM and Build Model

---

Fig. 1.Proposed K Means Cluster based oversampling Algorithm

Algorithm: Generate_Syntheticinstances

Input: Dec_instances[], SR, Dec_size
Output: Synthetic minority class instances

1. Let X_synth[]=empty
2. Let KN= Min(Dec_Size, 5)
3. For D=1 to Dec_size
    a) Let DK[]=Find KN of neighbors of Dec_instances [D]
    b) Let DK_size=number of instances in DK
    c) Let LSR= SR
    d) if (DK_size> SR) then
        I. Let LSR = SR / DK_size
    e) Else
        I. Let DK_size=1
    f) For L=1 to LSR
        I. For N=1 to DK_size
            i. Let delta = generate a randomnumber
            ii. Let synth = feature values Dec_instances [D] + (feature values of Dec_instances [D] – feature values of DK[N]) * delta
            iii. Append synth in X_synth[]
        II. Next
    g) Next
4. Next
5. Return X_synth[]

Fig. 2.Generate_Syntheticinstances Method

IV. EXPERIMENTAL RESULTS

In this section we have presented and discussed the experimental results achieved by our proposed method.

The proposed oversampling 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 datasets from [16] Keel data repository. 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	new-thyroid2	5	215	5.14
2	page-blocks0	10	5472	8.79
3	yeast3	8	1484	8.1
4	vowel0	13	988	9.98
5	yeast-1-2-8-9_vs_7	8	947	30.57
6	yeast-2_vs_4	8	514	9.08
7	yeast-2_vs_8	8	482	23.1
8	yeast5	8	1484	32.73
9	ecoli-0-6-7_vs_3-5	7	222	9.09
10	yeast-0-2-5-7-9_vs_3-6-8	8	1004	9.14

The number of features of the datasets range from 5 to 13. The number of instances in the datasets ranges from 215 to 5,472. The imbalance ratio of the datasets ranges from 5.14 to 32.73.

The dataset is partitioned into training (80%) and test datasets (20%). The test dataset is scaled with z-score based on the mean and standard deviation of training dataset. 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 oversampling 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 oversampling algorithms SMOTE, DBSMOTE and ADASYN. These popular oversampling algorithms are implemented using the "SMOTEfamily" package. Table II shows up the comparison results of AUC value of the algorithms.

The AUC comparison shows up that the proposed algorithm outperforms with SMOTE, DBSMOTE, ADASYN for all the 10 datasets. Table III shows up the comparison results of GMean value of the algorithms. The GMean comparison shows up that the proposed algorithm outperforms with all the three algorithms for the 10 datasets.

The proposed method performs better than the popular oversampling methods SMOTE, DBSMOTE and ADASYN. The proposed method outperforms in most of the datasets with imbalance ratio of 5.14 and above. The method identifies the class within sub concepts and then it generates synthetic instances based on the size of the class within sub concepts.

V. CONCLUSION

Imbalanced data classification problems present in many application areas like medical complication diagnosis, risk management, fault-detection, etc. It is an active research topic in the data mining and machine learning field. The proposed method is another novel method, added in the field. The proposed method balances the distribution by creating synthetic minority instances around the class within sub concepts and provides better results when comparing to the other existing methods. The future work is to adapt more suitable ensemble technique to enhance the performance of the algorithm.

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

S.No.	Dataset	Proposed Method	SMOTE	DBSMOTE	ADASYN
1	new-thyroid2	1	0.944444	0.944444	0.944444
2	page-blocks0	0.936721	0.835008	0.844759	0.805883
3	yeast3	0.931327	0.851576	0.918606	0.79491
4	vowel0	1	0.972222	0.972222	0.972222
5	yeast-1-2-8-9_vs_7	0.940413	0.547348	0.550157	0.538043
6	yeast-2_vs_4	0.955084	0.901572	0.937057	0.883516
7	yeast-2_vs_8	0.975	0.515278	0.526697	0.485714
8	yeast5	0.957386	0.859332	0.849395	0.859332
9	ecoli-0-6-7_vs_3-5	0.9875	0.820513	0.820513	0.820513
10	yeast-0-2-5-7-9_vs_3-6-8	0.931667	0.914697	0.870788	0.682847

Table- III: Performance Evaluation in terms to GMean with Proposed Oversampling algorithm and the Popular Oversampling Algorithms

S.No.	Dataset	Proposed Method	SMOTE	DBSMOTE	ADASYN
1	new-thyroid2	1	0.984251	0.984251	0.984251
2	page-blocks0	0.95114	0.931989	0.947265	0.94924
3	yeast3	0.929622	0.916569	0.870738	0.912076
4	vowel0	1	0.997308	0.997308	0.997308
5	yeast-1-2-8-9_vs_7	0.90438	0.575753	0.57755	0.568511
6	yeast-2_vs_4	0.906344	0.846886	0.851631	0.87956
7	yeast-2_vs_8	0.980581	0.523148	0.539656	0.439941
8	yeast5	0.95718	0.955134	0.917663	0.955134
9	ecoli-0-6-7_vs_3-5	0.816497	0.805961	0.805961	0.805961
10	yeast-0-2-5-7-9_vs_3-6-8	0.930233	0.927601	0.919662	0.841955

REFERENCES

1. I. Tomek, “Two modifications of CNN,” IEEE Trans. Systems, Man and Cybernetics, vol. 6, pp. 769–772, 1976.
2. 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.
3. 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.
4. Y. Dong and X. Wang, “A new over-sampling approach: random-SMOTE for learning from imbalanced datasets,” in International Conference on Knowledge Science, Engineering and Management, 2011, pp. 343–352.
5. S. Hu, Y. Liang, L. Ma, and Y. He, “MSMOTE: improving classification performance when training data is imbalanced,” in 2009 second international workshop on computer science and engineering, 2009, vol. 2, pp. 13–17.
6. C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, “Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem,” in Pacific-Asia conference on knowledge discovery and data mining, 2009, pp. 475–482.
7. H. Han, W.-Y. Wang, and B.-H. Mao, “Borderline-SMOTE: a new over-sampling method in imbalanced datasets learning,” in International conference on intelligent computing, 2005, pp. 878–887.
8. H. M. Nguyen, E. W. Cooper, and K. Kamei, “Borderline over-sampling for imbalanced data classification,” in Proceedings: Fifth International Workshop on Computational Intelligence & Applications, 2009, vol. 2009, pp. 24–29.
9. B. A. Almogahed and I. A. Kakadiaris, “NEATER: filtering of over-sampled data using non-cooperative game theory,” Soft Computing, vol. 19, no. 11, pp. 3301–3322, 2015.
10. J. De La Calleja and O. Fuentes, “A Distance-Based Over-Sampling Method for Learning from Imbalanced DataSets.,” in FLAIRS Conference, 2007, pp. 634–635.
11. H. He, Y. Bai, E. A. Garcia, and S. Li, “ADASYN: Adaptive synthetic sampling approach for imbalanced learning,” in 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 2008, pp. 1322–1328.
12. M. Koziarski, B. Krawczyk, and M. Woźniak, “Radial-Based oversampling for noisy imbalanced data classification,” Neurocomputing, vol. 343, pp. 19–33, 2019.
13. G. Douzas, F. Bacao, and F. Last, “Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE,” Information Sciences, vol. 465, pp. 1–20, 2018.
14. C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, “DBSMOTE: density-based synthetic minority over-sampling technique,” Applied Intelligence, vol. 36, no. 3, pp. 664–684, 2012.
15. I. Nekooeimehr and S. K. Lai-Yuen, “Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets,” Expert Systems with Applications, vol. 46, pp. 405–416, 2016.
16. J. Alcalá-Fdez et al., “Keel data-mining software tool: dataset 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.), Department of Computer Science, Madurai Kamaraj University, Madurai, Tamil Nadu, India.E-mail: gurusamyarumugam@gmail.com.