Ensembles for Class Imbalance Problems in various domains

N Deepakindresh
Vellore Institute of Technology, Chennai, India
deepakindresh.n2019@vitstudent.ac.in

Gauthum J
Vellore Institute of Technology, Chennai, India
gauthum.j2019@vitstudent.ac.in

Jeffrin Harris
Vellore Institute of Technology, Chennai, India
jeffrinharris.m2019@vitstudent.ac.in

Harshavardhan J
Vellore Institute of Technology, Chennai, India
harshavardhan.2019@vitstudent.ac.in

Abstract

The paper is an analysis of class imbalance problems from various domains such as medical, sentiment analysis, software defect, water potability and relationship status of students and summarizes the performance of data resampling techniques such as random undersampling, oversampling and synthetic minority oversampling technique and ensemble methods such as bagging, boosting and hybrid techniques used to solve the class imbalance problem.

1 Introduction

Dataset being the most important aspect of machine learning, without which no matter how strong or complex models are used it cannot solve a problem without having an abundant size of data. But real time data does not always promise this for all classes since only either class might dominate over the other class and hence lead to class imbalance problems where one class dominates over another and makes the smaller class look like noise in the dataset and misleads machine learning algorithms. This is a major problem as the models would generally underfit the data and if tweaked with a lot of hyperparameter tuning, it again quickly escalates to overfitting, which is why data resampling techniques are used where the data corresponding to the lower class are either randomly resampling or oversampling using nearest neighbours and the other method is under sample the higher classes. Both have their advantages and they are clearly analyzed in this paper. Apart from data resampling techniques another great way to counteract class imbalance problem is using ensemble models.

The ensembles for class-imbalance problem (ECIP) (Malhotra and Lata, 2020) are the modification of ensembles which pre-process the imbalanced data using data resampling before the learning process. This study experimentally compares the performance of several ECIP using performance metrics F1 score and g-Mean over five datasets which predicts heart failure rate[1], customer satisfaction[2], software defect[3], student relationship[4] and water potability[5]. Each of these datasets from 5 different domains have a class imbalance problem due to which normal stand alone models suffer from true positive identification which is identifying the lower class correctly.

2 Data Analytics

Data was collected from kaggle and it can be clearly seen from the graphs how big of a class imbalance problem is present in the dataset. For preprocessing columns with no correlation to the dataset were removed with a threshold set to less than 0.2 and rows with na values were removed.

For software defect dataset with features such as lines of code etc were used to predict the if the data has software defect or not, from the [Fig 1] of class imbalance problem is clearly visible where the system with no defects is the higher class and with defects is the class with very less data. There is an approximate of 11.60988948387832% of class imbalance in this dataset.

For Students dataset (Libbey, 2004) the response variable is the relationship status of the student which is predicted using features such as age, gender, marks, parents occupation etc and as seen from the [Fig 5] the percentage ratio of single to committed is 24.334600760456272%.

[1] https://www.kaggle.com/fedesoriano/heart-failure-prediction

[2] https://www.kaggle.com/teejmahal20/airline-passenger-satisfaction?select=train.csv

[3] https://www.kaggle.com/semustafacevik/software-defect-prediction

[4] https://www.kaggle.com/ramontanoeiro/student-performance/data

[5] https://www.kaggle.com/adityakadiwal/water-potability

For the Heart failure dataset various features were used to predict if the patient suffers from heart disease or not and the class imbalance ratio is 14.390243902439023%. The datasplit can be seen in [Fig 2].

For water potability (Itah and Akpan, 2005) dataset the features used where chemical quantities such as ph values etc and class imbalance is clearly visible from [Fig 4] where the imbalance ratio is 16.216216216216218%.

For the airline (An and Noh, 2009) dataset the features were ratings on each individual aspects such as reviews on maintenance, safety, communication, food quality etc were used to predict the total review of which was good or bad. The class imbalance ratio is 19.865504700473 48% and is visualized in [Fig 3].

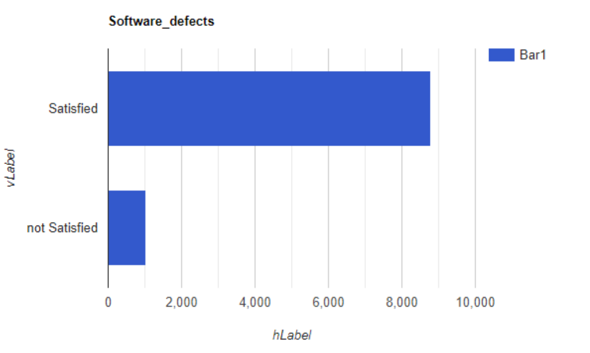

Figure 1: Distribution of defect and no defect class labels

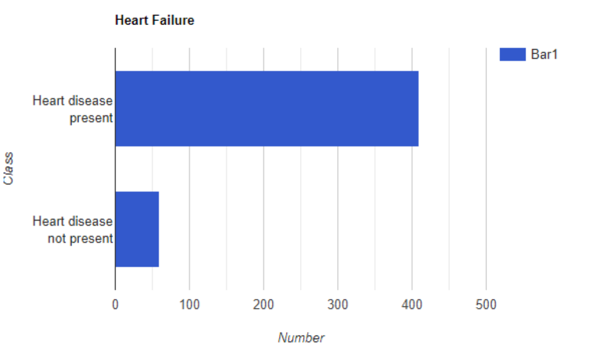

Figure 2: Distribution of disease and no disease class labels

3 Classic Ensembles

There are two sections namely the Classic ensembles and Ensembles for the class imbalance problem. The Classic ensembles employed in this research are briefly described in the first section, and the ECIP is described in the next section.

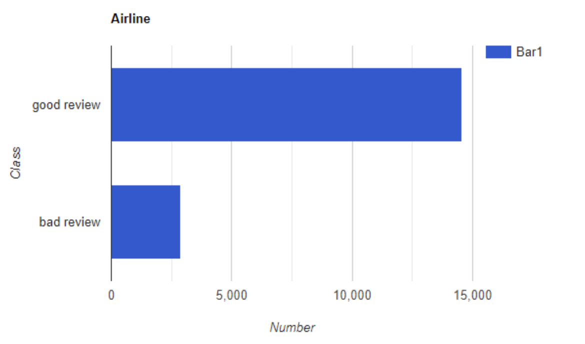

Figure 3: Distribution of good and bad review class labels

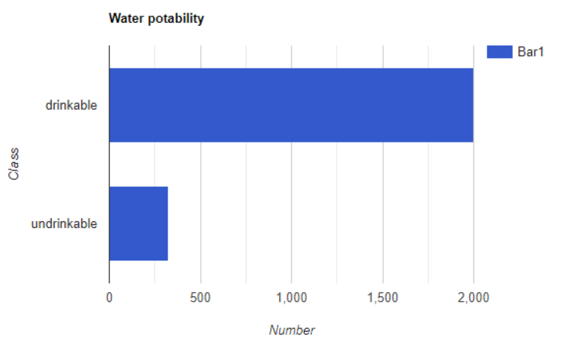

Figure 4: Distribution of drinkable and not drinkable class labels

3.1 Bagging

Bootstrap Aggregation (also known as Bagging) is a basic yet effective ensemble approach. The Bootstrap approach is applied to a high-variance machine learning system, such as decision trees (Pedregosa et al., 2011), in the process of bagging. Consider the case when there are N observations and M features .A sample from observation is selected randomly with replacement(Bootstrapping). A subset of features are selected to create a model with a sample of observations and subset of features. The feature from the subset that yields the best split on the training data is chosen. This process is repeated to produce a large number of models, each of which is trained in parallel. A forecast is made based on the sum of all of the models' predictions. Bagging is the ideal approach if the single model's issue is overfitting. Boosting, on the other hand, does not assist to avoid over-fitting; in fact,

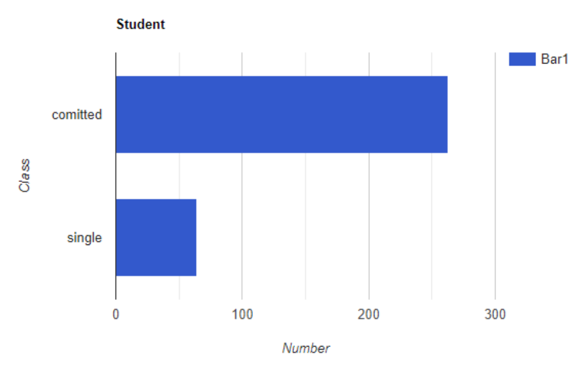

Figure 5: Distribution of in-relationship and not in relationship review class labels

this strategy is plagued by this issue. As a result, Bagging is more successful than Boosting in this situation.

3.2 Boosting

Boosting is a set of algorithms that use weighted averages to turn poor learners become good learners. Unlike bagging, which had each model run independently before aggregating the outputs in the end without giving any model preference. Boosting is all about ”collaboration.” Each model that runs determines which features will be prioritised in the following model. If the issue is that a single model performs poorly, Bagging is unlikely to produce a better bias. Boosting, on the other hand, may result in a combined model with reduced errors by maximising the advantages and minimising the drawbacks of the single model.

4 ECIP Model

The hybridization of data resampling techniques and classic ensembles is the ensemble for the class-imbalance problem. For the learning process, data resampling algorithms give balanced data distribution to ensembles.The baseline model that was used for these problems is Decision tree classifier which runs on non resampled data to differentiate the performance between normal models and ECIP models. The ECIP is categorized into three parts: boosting-based ensembles, bagging-based ensembles, and hybrid ensembles.

4.1 Boosting based ECIP Models

4.1.1 AdaBoost

AdaBoost is a boosting technique. The technique’s main focus is on the difficult-to-understand cases. The complete dataset is presented to each classifier sequentially at the start of the learning phase, with identical weights assigned to all of the instances. Following iteration, the algorithm’s main attention is on the cases that are difficult to classify, which are correctly categorised in succeeding iterations.

4.1.2 SMOTEBoost

SMOTEBoost is a combination of SMOTE and AdaBoost into a single method. With the aid of SMOTE, synthetic instances are produced in SMOTEBoost to oversample the minority class with each round of boosting. Each poor learner receives balanced material in this manner, allowing them to learn more effectively.

4.1.3 RUSBoost

RUSBoost is a combination of random undersampling and AdaBoost. It works similar to SMOTEBoost and MSMOTEBoost, but it ensures that each round of boosting has a balanced data distribution by removing the majority of class instances at random. RUSBoost is a simpler and quicker ensemble to deal with class-imbalance problems than SMOTEBoost and MSMOTEBoost since a lower amount of data points are provided to the classifiers during each round of boosting.

4.2 Bagging based ECIP Models

4.2.1 Over Bagging

This technique is a combination of oversampling sampling with Bagging. In these techniques,each subset of training data is generated by oversampling the instances of the minority class. The subsets of balanced training dataset are then used to train the individual classifiers in forming the ensemble.

4.2.2 Under Bagging

This technique is a combination of undersampling and bagging. Each classifier in the ensemble is built iteratively from a subset of a balanced training dataset obtained via undersampling, which contains the majority of class instances. When an unseen instance is provided to the ensemble after the individual classifiers have been constructed, the majority vote on the individual classifiers’ predictions is used to output the class label of the unseen instance.

S.No	Model	g-mean	F1-score
1	Decision Tree	0.3963	0.158590
2	Bagging	0.192285	0.069565
3	AdaBoost	0.096198	0.018182
4	Easy Ensemble	0.615819	0.202381
5	RusBoost	0.618666	0.227378
6	SmoteBoost	0.615819	0.202381
7	underBagging	0.556696	0.184332
8	Over Bagging	0.213254	0.069444

Table 1: Software Defects

S.No	Model	g-mean	F1-score
1	Decision Tree	0.915663	0.851518
2	Bagging	0.918186	0.890830
3	AdaBoost	0.865817	0.820467
4	Easy Ensemble	0.891907	0.740586
5	RusBoost	0.906828	0.828364
6	SmoteBoost	0.891907	0.740586
7	underBagging	0.928717	0.870704
8	Over Bagging	0.921145	0.885077

Table 2: Airline

5 Results

The results of the experiments with all the models mentioned above for the software defects, airline, heart disease, water potability, student relationship are provided at the tables below.

From the Table 1 we can see that RusBoost works with best efficiency for the Software Defects dataset followed by SmoteBoost and Easy Ensemble.

From the Table2 we can see that underbagging works with best efficiency for the Airline dataset followed by Over Bagging and Bagging.

From the Table3 we can see that SmoteBoost works with best efficiency for the Heart Disease dataset followed by Easy Ensemble and Over Bagging.

From the Table4 we can see that underBagging works with best efficiency for the Water potability dataset followed by Over Bagging and RusBoost.

From the Table5 we can see that RusBoost works with best efficiency for the Student Relationship followed by Under Bagging and Decision tree.

S.No	Model	g-mean	F1-score
1	Decision Tree	0.600481	0.480000
2	Bagging	0.693375	0.592593
3	AdaBoost	0.725563	0.500000
4	Easy Ensemble	0.764811	0.645161
5	RusBoost	0.762713	0.564103
6	SmoteBoost	0.764811	0.645161
7	underBagging	0.740322	0.666667
8	Over Bagging	0.745177	0.692308

Table 3: Heart Disease

S.No	Model	g-mean	F1-score
1	Decision Tree	0.915663	0.851518
2	Bagging	0.918186	0.890830
3	AdaBoost	0.865817	0.820467
4	Easy Ensemble	0.891907	0.740586
5	RusBoost	0.906828	0.828364
6	SmoteBoost	0.891907	0.740586
7	underBagging	0.928717	0.870704
8	Over Bagging	0.921145	0.885077

Table 4: Water potability

S.No	Model	g-mean	F1-score
1	Decision Tree	0.647382	0.357143
2	Bagging	0.551447	0.352941
3	AdaBoost	0.445904	0.235294
4	Easy Ensemble	0.572263	0.285714
5	RusBoost	0.709171	0.413793
6	SmoteBoost	0.572263	0.285714
7	underBagging	0.683986	0.434783
8	Over Bagging	0.450254	0.250000

Table 5: Student Relationship

5.1 Metrics and Evaluation

5.1.1 F1-Score

F1 Score (Huang et al., 2015) is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. The equation1 is provided below.

$$F1\_Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{1}$$

5.1.2 G-Mean

G-Mean (Espíndola and Ebecken, 2005) represents the geometric mean of true positive rate and true negative rate. A good prediction model should have high g-Mean. The equation 2 is provided below.

$$gMean = \sqrt{Recall * Specificity} \tag{2}$$

6 Conclusion

We can see that class imbalance problems are widespread in this current era of Big Data Collection. Implementation of efficient methods to counteract class imbalance problems is vital for progression in the machine learning domain. From our experiments we can clearly infer the best models for every domain consistently are RusBoost, UnderBagging followed with a little lesser performance from Overbagging, Easy ensemble and Smote Boost. As this experiment was conducted on datasets from multiple domains this could be a proof of concept that ECIP models will produce a better result for datasets having class imbalance problems as our best models generalize well for all types of data with various sizes.

Acknowledgments

We wish to express our sincere thanks and deep sense of gratitude to ourproject guide, Dr. Shivani Gupta, Associate Professor, School of Computer Science and Engineering (SCOPE), for her consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to Dr. Neela Narayanan, Dean of School of Computer Science and Engineering, VIT Chennai, for extending the facilities of the School towards our project and for his unstinting support. We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

References

Myungsook An and Yonghwi Noh. 2009. Airline customer satisfaction and loyalty: impact of in-flight service quality. *Service Business*, 3(3):293–307.

Rogério P Espíndola and Nelson FF Ebecken. 2005. On extending f-measure and g-mean metrics to multi-class problems. *WIT Transactions on Information and Communication Technologies*, 35.

Hao Huang, Haihua Xu, Xianhui Wang, and Wushour Silamu. 2015. Maximum f1-score discriminative training criterion for automatic mispronunciation detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(4):787–797.

Alfred Young Itah and Comfort E Akpan. 2005. Potability of drinking water in an oil impacted community in southern nigeria.

Heather P Libbey. 2004. Measuring student relationships to school: Attachment, bonding, connectedness, and engagement. *The Journal of school health*, 74(7):274.

Ruchika Malhotra and Kusum Lata. 2020. Using ensembles for class-imbalance problem to predict maintainability of open source software. *International Journal of Reliability, Quality and Safety Engineering*, 27:2040011.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.