International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 04 Issue: 08 | Aug -2017 www.irjet.net p-ISSN: 2395-0072

Handling Imbalanced Data: SMOTE vs. Random Undersampling

Satwik Mishra[1]

[1]*Department of I&CT*
Manipal Institute of Technology,Manipal, Karnataka - 576104, India

---

Abstract - *Imbalanced data refers to a problem where the number of observations belonging to one class is considerably higher than the other classes. This problem is predominated in cases where anomaly detection is of prime importance, for example fraud detection in banks, insurance etc. This paper aims to compare the results of different sampling methods namely Randomized Under Sampling, SMOTE with and without proper validation on a randomly generated imbalanced data set, with Random Forest and XGBoost as the underlying classifiers.*

Key Words: Imbalanced dataset, Random Undersampling, SMOTE, XGBoost, Random Forest, Cross Validation

1. INTRODUCTION

Machine learning algorithms are designed to reduce the error in order to improve the accuracy, in doing so; the distribution of classes is not taken into consideration. Hence in such cases it is extremely important to use apt performance metrics, sampling techniques and classifiers to get a better understanding of the data set and the distribution. Few important metrics are confusion matrix, precision, recall, ROC curves etc. Sampling techniques primarily are of two types Undersampling and Oversampling. This paper will compare the performance and results of classification done by the various combinations of classifiers, Random Forest and XGBoost and sampling techniques, Random Undersampling and SMOTE.

2. OBJECTIVE

When subjected to imbalanced data sets, machine learning algorithms face difficulties. The predictions made are biased and have misleading accuracy. This is due to lack of information about the minority class. The Machine Learning algorithms assume that data sets are balanced with equal class weights, and therefor tends to classify every test case sample into the majority class in order to improve the accuracy metric. The possible solution to this problem is the use of sampling techniques. For many base classifiers, studies have shown that a balanced data set provides a better overall classification as compared to that of an imbalanced data set [1].Hence the use of sampling methods on imbalanced data set is justified by these results. Nonetheless this does not imply that learning by classifiers cannot take place from imbalanced data set. Studies have shown that information gathered from certain imbalanced data is analogous to the same data treated under sampling techniques[2].The objective of this paper is to compare two important methods of handling this problem of imbalanced data, namely Randomized Undersampling and SMOTE, and their classification performance with two different classifiers, Random Forest and XGBoost.

3. METHODOLOGY

3.1 Data Sets

In our classification problem, the data set used is randomly generated so as to avoid any existing bias of the performance of one particular machine on a standard data set. The data set is divided into two classes and has 50000 samples. Located around the vertices of a hypercube in a 2 dimensional subspace, each class has equal number of Gaussian clusters within it. There are 20 features in a vector, of which 10 hold relevant or true information, 5 are redundant features, i.e. their values have no say on the classification of class to which a sample belongs to. Remaining 5 are not providing any additional information as they are highly positively correlated to some other features [3]. Python library imblearn is used to convert the sample space into an imbalanced data set. Ratio is set to 0.085 i.e. the ratio of number of samples in minority class to that of in majority class. Similarly functions such as RandomUnderSampler and SMOTE is used for desired sampling techniques available in the python library imblearn.

3.2 Random Forest and XGBoost

On combining various learning models, the classification accuracy increases: this is the basic idea behind bagging techniques. One of the most popular and powerful algorithms is Random Forest, which can perform both classification and regression. Random Forest works as a large collection of decorrelated decision tress. The term forest is there due to the use of many underlying trees. The more the trees the more is the precision. Multiple decision trees are built using algorithms like information gain or GINI index. On the basis of the vector input, each tree gives a classification by voting in favor of that corresponding class. If a classification problem, the forest then chooses the classification having the most votes or else it takes the average of all the trees if a regression problem [4].

XGBoost stands for eXtreme Gradient Boosting. It is an advanced implementation of gradient boosting. One of the most important features of this algorithm is parallel processing, thereby increasing the speed as compared to that of gradient boosting. Even if building of trees is sequential, one of the best ways to achieve parallel processing is 'Parallelize Split Finding at Each Level by Features'. XGBoost also has a built in cross validation, allowing the users to carry out validation at each iteration. Another important feature is regularization, helps preventing over-fitting [5].

3.3 Random Undersampling and SMOTE

Undersampling is one of the simplest strategies to handle imbalanced data. In Figure 1, the majority class, class 1 is undersampled. The blue and black data points represent class 1: blue dots are the removed sample, selected randomly from the majority class until the data is balanced. Removing data will obviously reduce the strain on storage and also improve run time. However, removing data might lead to loss of useful information.

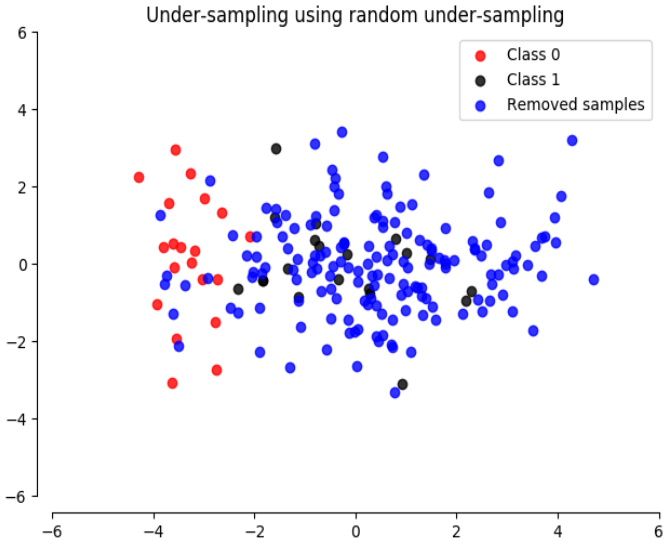

Figure 1 Graphical representation of random undersampling

In contrast to undersampling, SMOTE (Synthetic Minority Over-sampling TEchnique) is a form of oversampling of the minority class by synthetically generating data points. Corresponding to the amount of oversampling required, k nearest neighbors are chosen randomly [6]. Firstly, difference is taken between the sample under consideration and its corresponding nearest neighbors, multiplied by a random number generated between 0 and 1 and then finally adding this vector to the original vector under consideration [7]. However it is important to note that SMOTE cannot be directly applied on the entire data set, and then split the data into testing and training set. The paper discusses both the cases, that is with and without proper cross validation. If the data is first oversampled and then split into test and train, this will lead to misleading results, as in this case there is a high chance of same data being present in test as well as training set. So in order to avoid this, first the data is split into test and train, and the SMOTE is applied over the training data set for proper validation of the testing set. In Figure 2, on applying SMOTE the density of red dots increased in its vicinity.

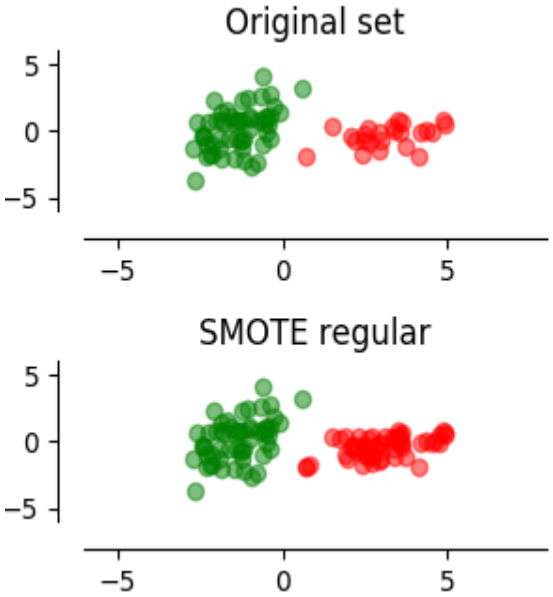

Figure 2 Original data vs. Oversampled Minority using SMOTE

3.4 Procedure

Once the data set is generated, using imblearn Python library the data is converted into an imbalanced data set. This imbalanced data set is then subjected to sampling techniques, Random Under-sampling and SMOTE along with Random Forest and XGBoost to see the performance under all combinations.

3.5 Model Evaluation

Statistical parameters such as ROC (Receiver Operating Characteristic) score, specificity and sensitivity is used to make comparison between the techniques used.

4. RESULTS AND DISCUSSION

With the help of all the ROC curves/ AUC (Area Under Curve), comparison can be easily made between various combinations of the techniques used. Figure 3 and 5 we can compare ROC under Random forest and XGBoost using random undersampling.

Figure 4 and 6 comparison is made between Random Forest and XGBoost with SMOTE with proper cross validation and without cross validation in Figure 7 and 8.

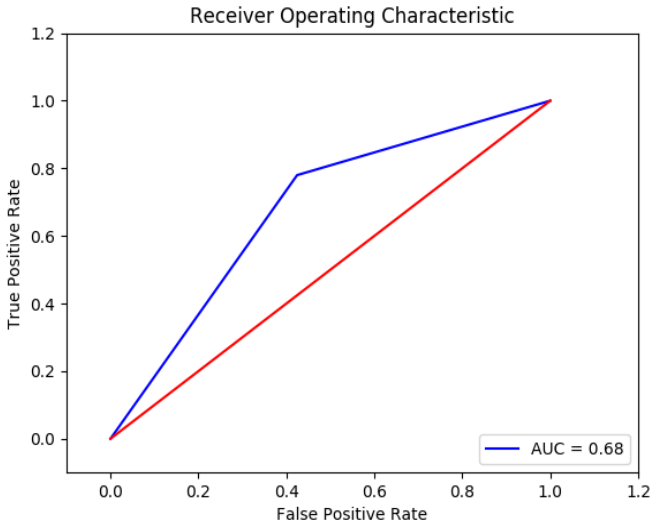

Figure 3: Random forest with random undersampling

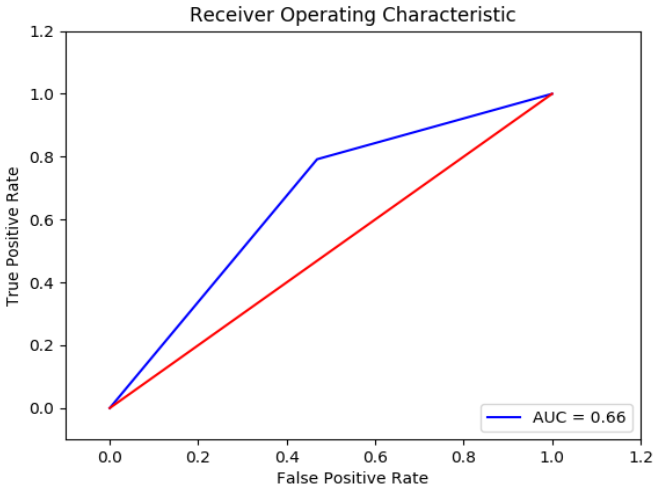

Figure 4: Random Forest with SMOTE with proper cross validation

ROC is a powerful tool to measure the performance of binary classifier. It is basically sensitivity vs. (1-specificity) graph. On the basis of the threshold or the cutoff decided by the classifier, sensitivity and the specificity is set, which will affect the true positives, true negatives, false positive and false negative. Hence ROC is a powerful measure to assess the prediction or classification made by the given machine learning algorithm. The more the area under the curve the better is the performance. ROC of 50% means, the classifier is not making any real classification, just predicting everything into one class, and 90%-100% means an excellent classifier.

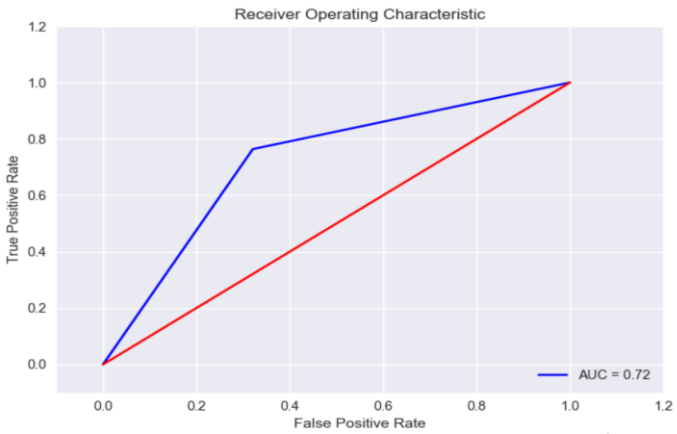

Figure 5: XGBoost with random undersampling

Figure 6: XGBoost with SMOTE with proper cross validation

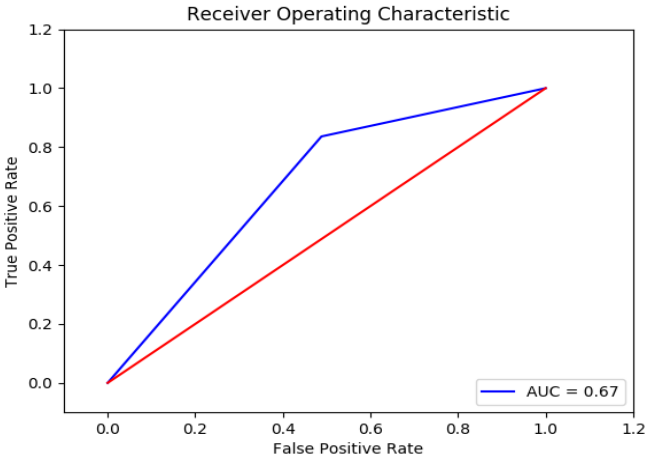

Figure 7: RF with SMOTE without proper cross validation

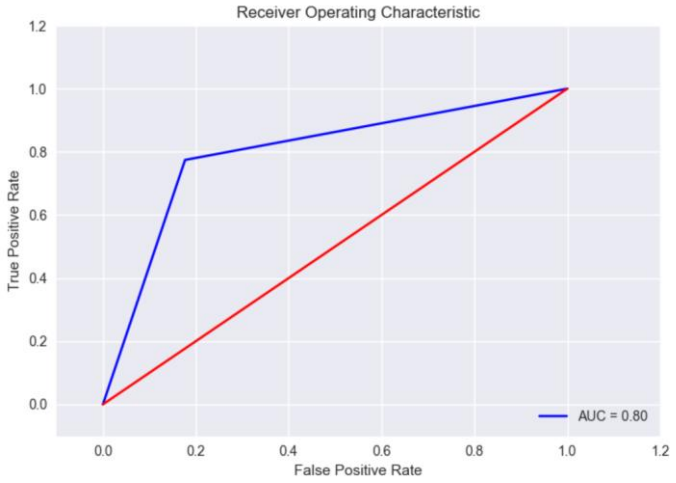

Figure 8: XGBoost with SMOTE without proper cross validation

	XGBoost		
	SMOTE with CV	SMOTE without CV	RUS
AUC %	67.59	79.87	72.15
Senstivity %	77.92	77.43	76.39
Specificity %	57.26	82.31	67.91

Figure 9: Table representing AUC, Sensitivity and Specificity for XGBoost

	Random Forest		
	SMOTE with CV	SMOTE without CV	RUS
AUC %	66	67	68
Senstivity %	79.19	83.62	77.99
Specificity %	53.12	51.19	57.56

Figure 10: Table representing AUC, Sensitivity and Specificity for Random Forest

5. CONCLUSION

This paper presents the results of subjecting imbalanced data to random undersampling and SMOTE and making classification using XGBoost and Random Forest. The research demonstrates the following:

- XGBoost performs better than Random Forest for all the combinations in terms of run time, tradeoff between sensitivity and specificity and ROC.
- ROC score under XGBoost is better than Random Forest for both the sampling techniques.
- XGBoost along with Random Undersampling gives the most balanced results with a good tradeoff between specificity and sensitivity.
- For the randomly generated data set, Random undersampling performed better than SMOTE under both the methods of classification, in terms of ROC score.
- XGBoost along with SMOTE without proper validation gives the best result numerically, however there is overfitting.
- With proper validation sensitivity is the highest for Random Forest along with SMOTE i.e. 79.19%.

6. REFERENCES

[1] G.M. Weiss and F. Provost, "The Effect of Class Distribution on Classifier Learning: An Empirical Study," Technical Report MLTR-43, Dept. of Computer Science, Rutgers Univ., 2001.

[2] G.E.A.P.A. Batista, R.C. Prati, and M.C. Monard, "A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data," ACM SIGKDD Explorations Newsletter, vol. 6, no. 1, pp. 20-29, 2004

[3] http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification.html

[4] RANDOM FORESTS, Leo Breiman Statistics Department University of California Berkeley, CA 94720 January 2001https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf

[5] Complete Guide to Parameter Tuning in XGBoost, Aarshay Jain, MARCH 1, 2016 Analytics Vidya

[6] https://www.jair.org/media/953/live-953-2037-jair.pdf

[7] Practical Guide to deal with Imbalanced Classification https://www.analyticsvidhya.com/blog/2016/03/practical-guide-deal-imbalanced-classification-problems/