ROSboost using COBRA for Imbalanced Class Problem

A Project Report Submitted
for the Course

MA691 Advanced Statistical Algorithms
Asst. Prof. Arabin Kumar Dey

by

Nishtha Sharma (180101052)
Aditya Patil (180101004)
Jay Chhajed (180123018)
Khushil Yadav (180103032)
Gauraangi Anand (180104032)

to the

DEPARTMENT OF MATHEMATICS
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, INDIA

November 2021

Abstract

The imbalance classification problem arises due to the unequal distribution of classes in the training dataset. This imbalance can vary from a slight bias to a severe imbalance. This imbalance is a huge challenge as most of the machine learning models are built considering an equal number of points for each class. Here we implement random oversampling boosting (ROSBoost) to solve the class imbalance problem. We have implemented boosting in a similar manner as AdaBoost with the oversampling of the minority. The weak classifier considered here is built using the COmBined Regression Alternative (COBRA). We observed an increase in minority class prediction with ROSBoost as compared to AdaBoost.

Table of Contents

Introduction

Imbalanced data can hamper our model performance big time. Class imbalance appears in many domains, including disease screening, spam filtering, and fraud detection. Resampling the data is one of the techniques used for solving this problem. Resampling can be further divided into oversampling and oversampling. Here we have used Synthetic Minority Oversampling Technique (SMOTE) for oversampling. Further, we have added boosting similar to AdaBoost to create ROSBoost which involves oversampling and training at each step and a weak learner is taken as a COBRA classifier.

Literature Survey

- Cobra [Biau et al., 2016]

  Cobra is a new method for combining several initial estimators of the regression function. Instead of generating a linear or convex optimized combination over a collection of estimators, they are used as a combined indicator of the nearness between a test observation and the training data. This local proximity approach is fast and model-independent. Moreover, the resulting combined estimator performs asymptotically (at least as well in the *L2* sense) as the best combination of the basic estimators in the collective.

  Given a set of preliminary estimators, the basic idea behind this combining method is the "unanimity" concept, which is based on the values predicted by the estimators for the data and for a new observation x. In summary, a data point is considered to be "close" to x, and consequently, dependable for contributing to the estimation of this new observation, if all estimators predicted values that were close to each other for x and this data item, meaning its not more distant than a predefined threshold ε. The predicted value corresponding to this query point x is then set to the average of the responses of the selected observations.

- AdaBoost [Schapire et al., 2013]

The AdaBoost algorithm was the first practical boosting algorithm, and remains one of the most widely used and studied, with applications in numerous fields.

It enables weak classifiers to enhance their performance by establishing the set of multiple classifiers. Also, since it automatically adapts to the error rate of the basic algorithm in training through dynamic regulation of the weight of each sample, a wide range of concern has been aroused.

Pseudocode for AdaBoost :

Given: $(x1,y1),\ldots,(xm,ym)$ where $xi \in X$, $yi \in \{-1,+1\}$. Initialize: $D1(i) = 1/m$ for $i = 1,\ldots,m$.

For $t = 1,\ldots,T$:

- Train weak learners using distribution *Dt* .
- Get weak hypothesis *ht* : X → {−1, +1}.
- Aim: select h_t with low weighted error:

$$\varepsilon_t = Pr_{i\sim D_t}[h_t(x_i) \neq y_i]$$

- Choose $\alpha_t = \frac{1}{2}ln(\frac{1-\varepsilon_t}{\varepsilon_t})$
- Update, for $i = 1,\ldots,m$:

$$D_{t+1}(i) = (D_t(i)exp(-\alpha_t y_i h_t(x_i)))/Z_t$$

where *Zt* is a normalization factor (chosen so that *Dt*+1 will be a distribution)

Output the final hypothesis:

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

- SMOTE [Chawla et al., 2002]

Barandela et al. discussed the problem of imbalanced training sets in supervised pattern recognition methods in their paper "*The Imbalanced Training Sample Problem: Under or over Sampling?"* One approach to addressing imbalanced datasets, as proposed by Kibler et al., is to oversample the minority class. The simplest approach involves duplicating examples in the minority class. Although it increases the amount of data, these examples don't add any new information to the model.

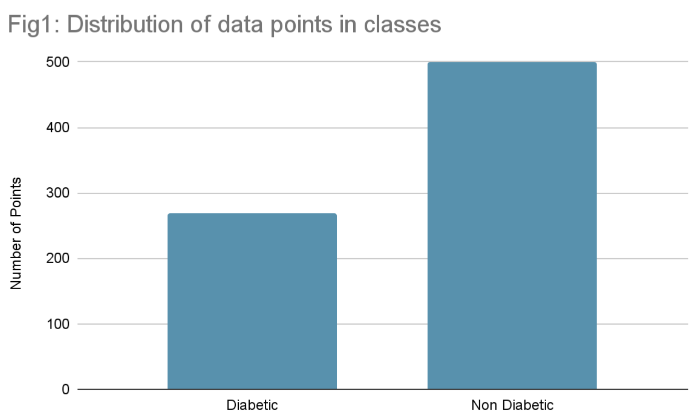
Instead, new examples can be synthesized from the existing examples. This is a type of data augmentation technique for the minority class and is referred to as the Synthetic Minority Oversampling Technique, or SMOTE for short. Chawla et al., proposed the under-sampling of the majority (normal) class to be a good means of increasing the sensitivity of a classifier to the minority class.

It works by utilizing a k-nearest neighbour algorithm to create synthetic data. SMOTE first starts by choosing random data from the minority class, then the k-nearest neighbours from the data are set. Synthetic data would then be generated from amongst the random data and the randomly selected k-nearest neighbour. The process is repeated enough times until the minority class has the same proportion as the majority class.

Dataset

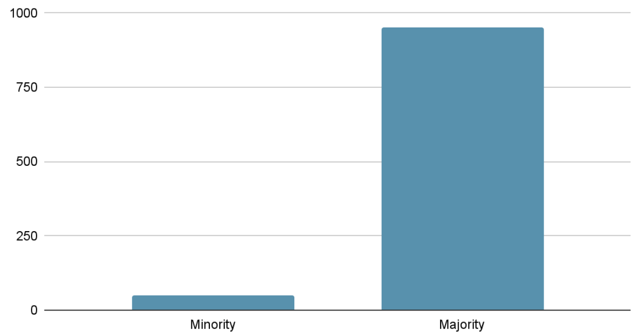
Two datasets were considered here:

- **Pima-indians-diabetes**: This dataset is from the National Institute of Diabetes and Digestive and Kidney Diseases. This can be used to predict whether a patient has diabetes based on certain diagnostic factors. The distribution of points in classes is shown in Fig1.



- **Generated Data**: We have used make_classification() scikit-learn function to create a synthetic binary classification dataset with 1000 points. Fig2 shows the distribution of points in classes.

Fig2: Distribution of data points in classes



Methodology

- We take the weak learners (of AdaBoost) to be Cobra Classifiers and consider an Imbalanced Dataset (D). We train the Cobra Classifier (C1) over the Dataset D.
- The points in minority class will be oversampled using the SMOTE algorithm, creating a Dataset D'..
- A new Cobra Classifier (C2) will be trained over this dataset D', similar to AdaBoost.
- Repeating the above process i.e Training, Oversampling, and Boosting, an ensemble of Cobra Classifiers will be trained, whose results will be aggregated using the weighted sum that Adaboost normally employs.

Metrics for Evaluation

Accuracy is not a good metric for the evaluation of classification in the case of an imbalanced dataset. Here consider the generated dataset which was described above. Here if the classifier only predicts the majority class it will get an accuracy of 95%. Hence we here consider confusion matrix, precision, recall, and MCC here.

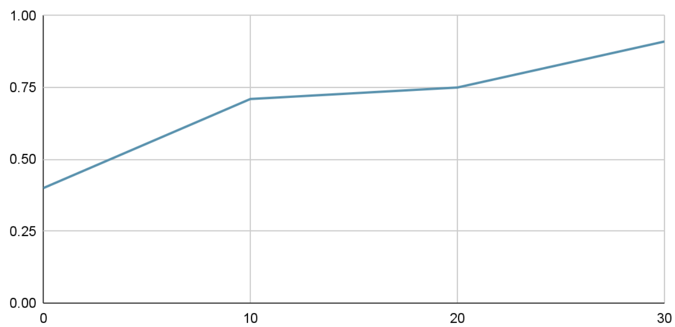
Results

Tables 1, 2, 3, and 4 show the results obtained. These results were obtained after running ROSBoost on a cobra classifier with 3 decision trees.
Here,
n_samples: number of points generated while oversampling
k_neighbours: number of neighbors to be considered in SMOTE.

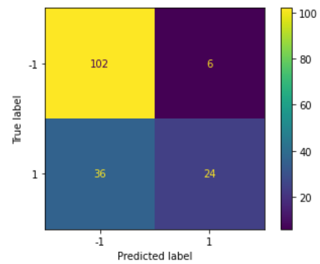
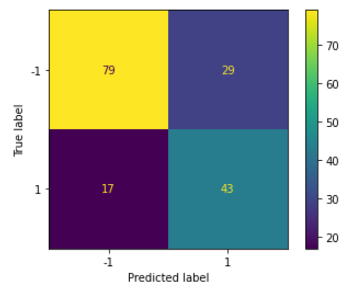
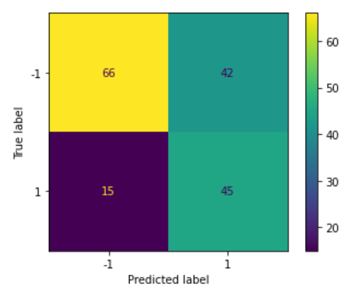
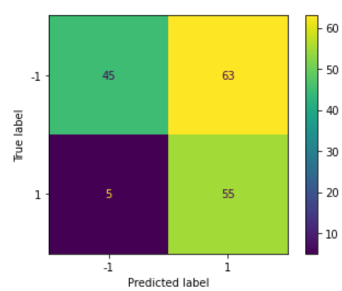
The case where n_samples is 0, is the performance on AdaBoost (ROSBoost without oversampling)

- Here on increasing n_samples to a certain point, we see an increase in the number of predictions made as minority class(class 1). Consider Table 3, here results for 0 and 50 n_samples is the same as oversampling done is not enough. On reaching 100 we can observe changes.
- On further increasing n_samples the number of predictions for minority class further increases which leads to deterioration of the performance by making wrong predictions for majority class. This is evident from Tables 1, 2, 3, 4 where n_samples are varied. On a very large value, all the predictions made were of the minority class.
- An increase in precision was also obtained on increasing n_samples which is shown in Fig3. Same observations were obtained from the generated dataset which is evident from Tables 3 and 4.

Fig3 Precision vs n_samples ,pima_indians-diabetes, k_neighbors=3



- Results on varying k_neighbours can also be compared. Table 1 and 3 contains results with k_neighbors=3 and table 2 and 4 contains results with k_neighbors=5. On a very large value of k_neighbors, a lot of noisy data is generated which leads to decreased quality of oversampling.

TABLE 1 **Pima-indians-diabetes**						
n_samples	k_neighbors	Accuracy Score	Recall	Precision	MCC	Confusion Matrix
0		0.75	0.8	0.4	0.43	
10	3	0.73	0.60	0.72	0.43	
20	3	0.66	0.52	0.75	0.35	
30	3	0.60	0.47	0.92	0.35	

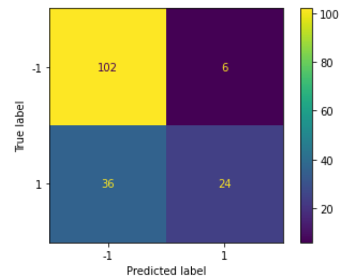
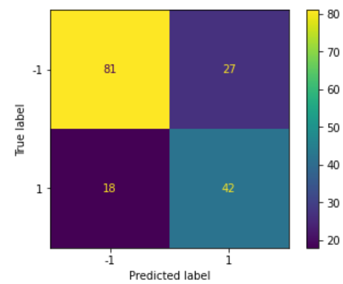
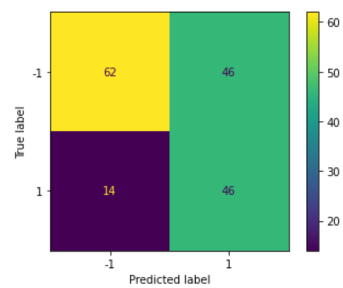
<table>
<tr><th colspan="7">TABLE 2 Pima-indians-diabetes</th></tr>
<tr><th>n_samples</th><th>k_neighbors</th><th>Accuracy Score</th><th>Recall</th><th>Precision</th><th>MCC</th><th>Confusion Matrix</th></tr>
<tr><td>0</td><td></td><td>0.75</td><td>0.8</td><td>0.4</td><td>0.43</td><td>
</td></tr>
<tr><td>10</td><td>5</td><td>0.73</td><td>0.61</td><td>0.7</td><td>0.44</td><td>
</td></tr>
<tr><td>20</td><td>5</td><td>0.64</td><td>0.5</td><td>0.77</td><td>0.33</td><td>
</td></tr>
</table>

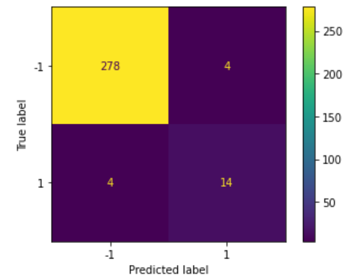
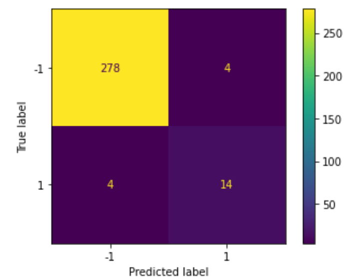
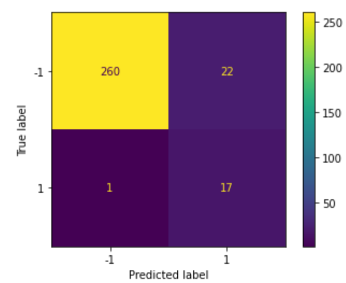
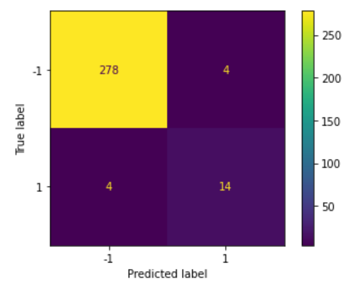
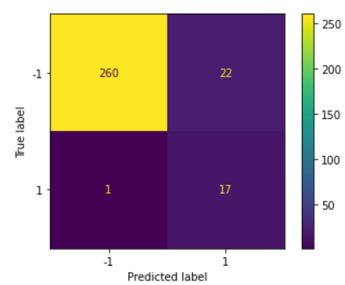
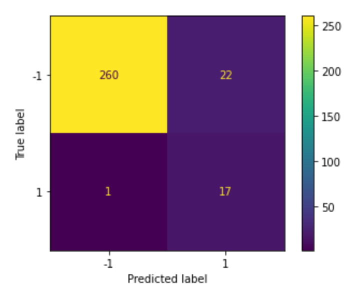
TABLE 3 Generated Data						
n_samples	k_neighbors	Accuracy Score	Recall	Precision	MCC	Confusion Matrix
0		0.97	0.78	0.78	0.76	
50	3	0.97	0.78	0.78	0.76	
100	3	0.92	0.44	0.94	0.61	

TABLE 4 Generated Data						
n_samples	k_neighbors	Accuracy Score	Recall	Precision	MCC	Confusion Matrix
0		0.97	0.78	0.78	0.76	
100	5	0.92	0.44	0.94	0.61	
150	5	0.92	0.44	0.94	0.61	

Conclusion

While boosting results in better performance in most of the cases there was no method to handle the class imbalance problem in each step. Here we implemented the same. On comparing results with AdaBoost (case in the table where n_samples=0) there was an increase in predictions for minority class as n_samples were increased. On a very large n_samples, all the predictions made were of the minority class. On low n_samples the result was the same as that of AdaBoost. Also the choice of k_neighbors is also important so that noisy data is not added in oversampling.

References

[Biau et al., 2016] Biau, G., Fischer, A., Guedj, B., and Malley, J. D. (2016). Cobra: A combined regression strategy. *Journal of Multivariate Analysis*, 146:18–28.

[Chawla et al., 2002] Chawla, Nitesh V., et al. "SMOTE: synthetic minority over-sampling technique." *Journal of artificial intelligence research* 16 (2002): 321-357.

[Santoso et al., 2019] Santoso, N., W. Wibowo, and H. Hikmawati. "Integration of synthetic minority oversampling technique for imbalanced class." *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)* 13.1 (2019).

[Schapire et al., 2013] Schapire, Robert E. "Explaining adaboost." *Empirical inference*. Springer, Berlin, Heidelberg, 2013. 37-52.