International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 3 Issue: 10 6049 - 6053

Machine Learning for Classification of Imbalanced Big Data

Takshak Desai
Department of Computer Engineering
D. J. Sanghvi College of Engg.
Mumbai, India
takshakpdesai@gmail.com

Udit Deshmukh
Department of Computer Engineering
D. J. Sanghvi College of Engg.
Mumbai, India
udit2594@gmail.com

Prof. Kiran Bhowmick
Department of Computer Engineering
D. J. Sanghvi College of Engg.
Mumbai, India
kiran.bhowmick@djsce.ac.in

***Abstract*—** The problem of classification of imbalanced datasets is a critical one. With an increase in the number of application domains that rely on classification, extensive research has been carried out in this field; with focus directed towards the problem of poor classification accuracy. Of late, the rise in significance of Big Data has forced industries to search for better techniques to handle massive and unstructured datasets; this has led to a need for robust classification algorithms that deal with unbalanced Big Data. This paper surveys the current algorithms provided by Machine Learning for unbalanced dataset classification and considers their possible use for larger or unstructured datasets.

Keywords-*unbalanced datasets;classification;machine learning;Big Data;unstructured data*

*****

I. INTRODUCTION

Imbalance in datasets typically refers to the non-uniform distribution of classes among its records. The distribution is highly uneven: more than 90% of the records belong to a particular class (the majority or negative class) while the remaining records belong to the other class (the minority or positive class). Standard algorithms used for classification in data mining assume a bias towards the majority class as they assume that the dataset they are working on is stratified and hence show poor accuracy when classifying unbalanced data. Further, these algorithms are accuracy driven i.e. they are concerned with minimizing the overall error to which a very small contribution is made by the minority class [3].

The problem gets worse when unbalance occurs in Big Data as a large number of records could get misclassified. It could lead to disastrous problems in cases such as medical diagnosis when an otherwise malignant tumor would get classified as harmless. Thus, extensive research has been carried out in this field as the number of application domains that rely on classification is increasing gradually. Broadly, solutions have been suggested at both the data level and the algorithm level to combat the problem. At the data level, solutions include different forms of sampling. At the algorithm level, one-class learning and cost-sensitive classification are suggested to counter the imbalance in data.

This paper surveys the existing algorithms used for classification, analyses the modifications made in them to cater to the imbalance in datasets and observes the practicality of these algorithms if the imbalance were to occur in Big Data. It concludes by recommending suitable areas for further research.

II. EXISTING ALGORITHMS

A. *Decision Tree*

Decision trees are one of the most popular classification algorithms used in data mining. Formally, it can be defined as a labelled tree where each interior node is labeled with an attribute, each of the leaf nodes in labeled with a class and each edge directed from an interior node is labeled with a value of range of values of the attribute at that node.

A decision tree construction algorithm (e.g. ID3, C4.5) commences with the root node that represents all the records present in the training set. It recursively partitions these records into nodes; for each partition, a child being created to represent it. The split into partitions is determined by the values of an attribute called the splitting attribute. Common examples of the criteria used to determine the splitting attribute include information gain, gini value, etc. The recursive splitting terminates at nodes that are pure i.e. all the records in these nodes belong to one class only. In some cases, it may also terminate at nodes that are almost pure i.e. the records mostly belong to a particular class.

A decision tree can be visualized as a graphical model of the dataset that can be used to predict the class for new records. For any new record, the algorithm that predicts the class of a record starts at the root node and at each edge, a decision is taken whether to traverse along that edge or not. When a leaf node is reached, the class labelled at that node is output as the class of the record to be classified.

A decision tree is typically evaluated by predictive accuracy that considers all errors equally. However, predictive accuracy might not be appropriate when the data is unbalanced or if the costs of the errors differ markedly. The probability that some branches that predict the small classes are removed and the new leaf node is labelled with a dominant class is very high. Further, if the training set is unbalanced, decision trees may have to create several tests to distinguish the small classes from the dominant classes [1] [2]. Thus, if unbalanced data sets are used to construct the decision tree, it may lead to inflated performance estimates: this will generate false conclusions for the record whose class is to be determined.

B. *Associative Classifier*

Associative Classifiers are techniques that make use of association rule mining to carry out the classification of records. These techniques are based on the assumption that each record contains only Boolean attributes and a class label. This assumption is not restrictive as categorical and qualitative attributes can be converted into Boolean attributes. e.g. For a numeric attribute, the value of the attribute for a record lies within a certain range or not can indirectly make the attribute function as a Boolean attribute.

Once this modification has been made, association rule mining can be carried by implementing any standard frequent itemset mining algorithm. Since the objective of an associative classifier is the classification of records, we are interested in

mining only those rules where the RHS consists of a class label i.e. for a class association rule X→C, if X is in a test record, the class is likely to be C. In the event of a conflict i.e. when two association rules are mined such that one rule classified a record to class C1 and the other to class C2; the rule with higher confidence value can be chosen.

For an unbalanced data set, association rules describing smaller classes are not likely to be found as such rules may have low confidence values or because patterns describing small classes may have a rare occurrence in the dataset.

C. *Neural Networks*

Neural Networks use a technology that attempts to produce intelligent behavior by trying to mimic the structure and functions of the nervous system. The nervous system is usually abstracted as a weighted directed graph where nodes function as neurons and the values associated with the links between these nodes indicate the type and strength of association. Neural networks are extremely popular in pattern recognition and machine learning applications.

Back propagation algorithms provide the best neural network architecture for classification problems. Experimental studies on training unbalanced bi-class data sets revealed that the net error for samples in the majority class got reduced quickly in the first few iterations but the net error for minority class increased considerably. Theoretical analysis indicated that this phenomenon occurred because the gradient vector computed by the standard backpropagation algorithm was in a downhill direction for the majority class but in an uphill direction for the minority class. This is because the length of the gradient vector of the majority class will be much larger than that of the minority class. Therefore, the gradient vector is dominated by that of the majority class. Consequently, if weights are recalculated in the direction of the gradient vector, the net error of the majority class will decrease significantly and that of the minority class will increase significantly. Empirical studies also reported that the subsequent rate of decrease of net error for the small class was very low. It needed thousands of iterations to reach an acceptable solution [1] [2].

D. *Support Vector Machines (SVMs)*

SVM refers to a supervised learning model which categorizes data points into two distinct classes by defining an optimum hyperplane between the two classes. An optimum hyperplane is the one which has the largest margin. Margin is the minimum distance between two data points belonging to different classes. SVMs seek an optimal separating hyperplane, where the margin is maximal. The solution is based only on those data points at the margin. These points are called as support vectors. The linear SVMs have been extended to nonlinear examples when the nonlinear separated problem is transformed into a high dimensional feature space using a set of nonlinear basis functions. However, the SVMs are not necessary to implement this transformation to determine the separating hyperplane in the possibly high dimensional feature space. Instead, a kernel representation can be used, where the solution is written as a weighted sum of the values of a certain kernel function evaluated at the support vectors. When perfect separation is not possible, slack variables are introduced for sample vectors to balance the tradeoff between maximizing the width of the margin and minimizing the associated error.

SVM assumes that only support vectors are informative in the process of classification and rest of the data points are redundant. However, in unbalanced data set, the majority class pushes the hyperplane closer to the minority class. Due to this, the support vectors of the majority class may be redundant and more informative data points may hide behind them [1] [8].

E. *K-Nearest Neighbours Classifier*

KNN is one of the simplest machine learning algorithms to be used for classification. In a bi-class classification problem, the Euclidian distance between the testing data point and the data points in the training set is computed and the k-nearest neighbors of the testing point are found. Depending on the class of the majority neighbors, the testing point is classified as the class of the majority neighbors. It is crucial to set the optimum value of k. A smaller value of k will lead to overfitting while a larger value may lead to under-fitting. To set the optimum value of K, k-fold cross validation is used.

The drawback of using KNN is that it is computationally expensive to find the k nearest neighbors; and hence for a large data set, it is not feasible at all to use KNN [3] [12]. Further, along with the size of the data set, even the dimensions of features adds a computational cost. Feature extraction hence becomes crucial.

F. *Naïve Bayes Classifier*

Naïve Bayes classifier is a probabilistic classifier. It is based on Bayes' theorem with independence assumptions between predictors. Bayes theorem provides a way of calculating the posterior probability, P(c|x), from P(c), P(x), and P(x|c). Naive Bayes classifier assume that the effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors. This assumption is called class conditional independence.

P(c|x) is the posterior probability of class (target) given predictor (attribute). P(c) is the prior probability of class. P(x|c) is the likelihood which is the probability of predictor given class. P(x) is the prior probability of predictor.

$$P(c|x) = P(x|c).P(c) / P(x) \quad (1)$$

$$P(c|X) = P(x1|C).P(x2|C)\ldots P(xn|C).P(c) \quad (2)$$

The advantage of using this classifier is that it simple to calculate. If the NB conditional independence assumption actually holds, a Naive Bayes classifier will converge quicker than discriminative models like logistic regression, so you need less training data. And even if the NB assumption doesn't hold, a NB classifier still often performs surprisingly well in practice. However, for a given unbalanced data set, dependency patterns inherent in the small classes are usually not significant and hard to be adequately encoded in the networks. When the learned networks are inferred for classification, the samples of the small classes are most likely misclassified [1] [3].

III. TOWARDS BIG DATA

Big Data collectively refers to those datasets that are so massive and complex that traditional algorithms cannot handle them adequately. Particularly, when it comes to mining meaningful data from such complex datasets, there is a need for powerful processors and analytic skills that successfully handle such data. While many technologies and techniques have been proposed to handle Big Data; a large part of Big Data remains grossly underutilized and unexplored. This is largely due to the unstructured nature of Big Data i.e. content that does not

conform to a specific model. With the increase in number of sources that contribute to the volume of Big Data, it would not be wrong to estimate that a large part of this data would be unstructured. Thus, any research efforts that are dedicated towards managing Big Data are majorly focused on the massive and unstructured nature of data that is increasing with a rapid velocity.

Machine learning literature on this topic has provided several solutions to cater to this problem. We discuss these solutions in the following section.

IV. HANDLING BIG DATA

A. *Sampling*

One solution is to carry out the undersampling of the majority class or oversampling of the minority class. If oversampling of the smaller class is carried out, typically through duplication of records, the decision region for the smaller class can become very specific i.e. it can lead to overfitting of the training data. A popular oversampling algorithm called SMOTE [11] adds new, non-replicated smaller class records to the training set. Undersampling of the dominant class by removing random records from the dominant class can also cause the smaller class to have a larger presence in the training data set. This technique can provide potential results with techniques such as decision trees, associative classifiers and Naïve Bayes classifier [11].

However, constructing a decision tree for a very large dataset is inadvisable as searching for meaningful areas within a very large decision tree is time consuming [6]. A solution to this problem is to build a set of decision trees in parallel on smaller datasets that are subsets of the original training set. The trees constructed are then reduced to a set of rules, any conflicting rules are resolved and then merged together to generate one set that is used for classifying the new records. The general strategy followed is to partition the training set into n subsets and apply the decision tree algorithm discussed earlier on each subset independently and in parallel. The output of this step will be independent decision trees that model the partitions. These independent decision trees are then converted to decision rules [5].

Combining the decision rules into a unified rule will not suffice as two rules may conflict each other i.e. one rule may classify a record as class ‘A’ while the other may classify the same record as class ‘B’. One technique for combining decision trees is where conflicts could be resolved by considering the conditional tests used to generate the rule. By modifying the conditional tests, one of the two rules can be chosen or the two rules can be modified to remove the ambiguity [5]. The final rule obtained will model the entire dataset and can be converted to a decision tree. Using this decision tree, the class for the new records can be predicted using the decision tree algorithm as discussed previously.

In the case of unbalanced big data, the partitioning of data should take place in a manner such that the partitions mirror the distribution of classes in the original dataset. E.g. If the original dataset comprises 98% records of class A and 2% records of class B; then the partitions should also possess the same or marginally different ratio of records. This could remove any classification inaccuracies and prevent misclassification of new records.

B. *Cost sensitive learning*

An alternative to sampling is to make use of cost-sensitive learning that takes into account the miscalculation costs associated with the dataset being classified. Cost-sensitive learning assigns a higher cost whenever the record belonging to minority class is misclassified, as compared to a majority class example being misclassified. A cost-sensitive learning technique takes into account the cost matrix of the classifier in the design phase and selects the model with the least cost. A possible solution to improve classification accuracy of an algorithm would be to first run the algorithm on the dataset, obtain the confusion matrix and make use of this confusion matrix to obtain the cost.

A major drawback of using cost-sensitive learning is that the cost matrix for a dataset is often unavailable; thus giving sampling a preference over cost-sensitive learning. However, empirical studies [16] have revealed that for datasets containing more than 10000 records, cost-sensitive learning is arguably a better option when compared to sampling. Thus, for an unbalanced big data set, cost-sensitive learning would prove to be a better alternative.

C. *Converting unstructured data to structured data*

Gartner’s report [14] revealed that up to 80% of Big Data is unstructured. Thus, it becomes imperative to take into account the unstructured nature of Big Data while using any classification algorithm.

To handle unstructured data, a decision tree algorithm called CUST [7] was designed. CUST introduces the use of splitting criteria formed by unstructured attribute values and reduces the number of scans on the datasets by using appropriate data structures. The first step involved is to convert the unstructured dataset to a structured dataset. E.g. If the dataset consists of “Website Content” as an attribute, then it can be converted to a structured attribute by representing each text content of the webpage as a Boolean attribute. Then any of the decision tree algorithms can be applied for classification.

Similarly, any other algorithm can be applied to unstructured datasets after converting these unstructured datasets to structured datasets. In case of Big Data, this would obviously lead to an increase in the size of the data used. In that case, associative classifiers would be a better option. Associative classifiers [3] can directly handle unstructured data, and are able to mine more meaningful rules especially when compared to the decision tree classifier algorithm. But this algorithm is very slow to work with.

D. *Using GSVM-RU and DC-SVM*

To address the challenges posed by big data, a novel divide and conquer approach (DC-SVM) to efficiently solve the kernel SVM problem has been proposed. DC-SVM achieves faster convergence speed compared to state-of-the-art exact SVM solvers, as well as better prediction accuracy in much less time than approximate solvers. To accomplish this performance, DC-SVM first divides the full problem into smaller sub problems, which can be solved independently and efficiently. It is theoretically shown that the kernel k means algorithm is able to minimize the difference between the solution of sub problems and of the whole problem, and support vectors identified by sub problems are likely to be support vectors of the whole problem. However, running kernel k means on the whole dataset is time consuming, so a

two-step kernel k means procedure to efficiently find the partition is applied. In the conquer step, the local solutions from the sub problems are glued together to yield an initial point for the global problem. The coordinate descent method in the final stage converges quickly to the global optimal.

The data set is divided into k sub problems. The quadratic optimization problem is solved for all the k sub problems independently. The solution of each sub problem is then combined. Since the set of support vectors for all the sub problems is expected to be close to the support vectors of the entire problem, the co-ordinate descent solver converges quickly.

Since the data set of concern is unbalanced, it must be ensured that each sub problem must contain the same ratio of positive to negative samples, as in the original data set. After dividing the data set into k subsets, GSVM-RU is applied to each subset. The process of GSVM-RU is described below.

SVM assumes that only support vectors are informative in the process of classification and rest of the data points are redundant. However, in unbalanced data set, the majority class pushes the hyperplane closer to the minority class. Due to this, the support vectors of the majority class may be redundant and more informative data points may hide behind them [10].

A single SVM cannot guarantee to extract all the informative support vectors at once [8]. However, it is safe to assume that a single SVM can extract at least a subset of informative support vectors. Based on this assumption, multiple information granules containing different informative samples can be formed using the technique of granulation. Initially, all the positive samples are considered informative and included in the positive informative granule. The negative samples contributing to the SVM are considered to be included in the negative information granule. This negative granule is then removed and another SVM is constructed. The negative samples contributing to the SVM are then added to the second negative information granule. The second negative information granule is then removed and the process is repeated. The negative samples which are not included in any information granule are discarded. Finally, the negative information granules are aggregated with the positive granule to construct the final SVM [9].

Then aggregation takes place. The aggregation dataset is initialized to the positive samples. The performance criteria is initialized to be the performance of the basic SVM. The first negative information granule is aggregated with the aggregation set. A new SVM is constructed and the performance is measured. If the performance is better than before, aggregation is continued by adding the second negative information granule in the aggregation dataset. The process is continued until the performance of the new classifier is inferior to the previous one.

GSVM-RU reduces information loss by including only the support vectors during each phase of granulation. As opposed to information loss, it leads to cleaning the dataset [9].

E. Modifying KNN for unstructured datasets

An experiment conducted on the classification of text documents [15] made use of an improved rule called NWKNN. It was found to yield better performance than the traditional KNN.

V. PERFORMANCE EVALUATION

When evaluating the performance of the classifier, accuracy could be replaced by balanced accuracy. The definition of accuracy is obtained from the normalized confusion matrix designed for a classifier. We define a as the number of true positives, b as the number of false positives, c as the number of false negative and d as the number of true negatives.

For a bi-class problem, accuracy would be defined as:

$$\text{Accuracy} = a+d / (a+b+c+d) \quad (3)$$

Balanced accuracy would be defined as the arithmetic mean of class specific accuracies.

$$\text{Balanced accuracy} = 0.5\,(a / a+b) + 0.5\,(c / c+d) \quad (4)$$

The use of balanced accuracy has found to avoid inflated performance estimates on the datasets. [1] [2]

Some classifiers e.g. Naïve Bayes classifier and Neural Networks make use of AUC or Area Under Curve as a performance parameter. Two parameters are defined as:

$$\text{True Positive Rate (TPR)} = a / (a+b) \quad (5)$$
$$\text{False Positive Rate (TFR)} = c / (c+d) \quad (6)$$

A curve called the ROC (Receiver Operating Characteristics) curve is plotted with TPR on Y-axis and TFR on X-axis. The area under this curve is abbreviated as AUC. This provides a single measure of a classifier's performance for evaluating which model is better on average even if the classifier works on an unbalanced dataset [3] [13].

Another widely used measure for finding the accuracy of a classifier are the F1-measure and G-score. F1-measure is obtained from the confusion matrix as:

$$\text{Precision} = a/a+b \quad (7)$$
$$\text{Recall} = a/a+c \quad (8)$$
$$\text{F1-measure} = 2\,(\text{precision}.\text{recall})/(\text{precision}+\text{recall}) \quad (9)$$

F1-measure finds application in the field of natural language processing, machine learning and information retrieval.

Similarly, G-score is defined as the geometric mean of precision and recall. It finds application in computational linguistics; and is very popular: many software have predefined methods for calculating the G-score.

VI. CONCLUSIONS AND FURTHER RESEARCH

SVMs are reported to be least affected by class imbalance problems. It has also been empirically observed that they show the best classification accuracy when used for classifying unbalanced data [1]. Hence, for massive datasets, SVMs would prove to be the best classification algorithm. An unstructured Big Data set could be converted to a structured dataset before using the SVM for classification.

However, the need to develop an algorithm or classification method that handles unstructured data directly as is the case with associative classifiers (Which are slow to work with) is clear. Research is currently being carried out in the direction of

neural networks, fuzzy logic and genetic algorithms to build a classifier that can work on unstructured data directly

REFERENCES

[1] Sun, Yanmin, Andrew KC Wong, and Mohamed S. Kamel. "Classification of imbalanced data: A review." International Journal of Pattern Recognition and Artificial Intelligence 23, no. 04 (2009): 687-719.

[2] Han, Jiawei, Micheline Kamber, and Jian Pei. Data mining: concepts and techniques: concepts and techniques. Elsevier, 2011.

[3] Ganganwar, Vaishali. "An overview of classification algorithms for imbalanced datasets." International Journal of Emerging Technology and Advanced Engineering 2, no. 4 (2012): 42-47.

[4] Chawla, Nitesh V. "C4. 5 and imbalanced data sets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure." In Proceedings of the ICML, vol. 3. 2003.

[5] G.J. Williams, "Inducing and Combining Multiple Decision Trees", PhD Thesis, Australian National University, Canberra, Australia, 1990.

[6] Hall, Lawrence O., Nitesh Chawla, and Kevin W. Bowyer. "Decision tree learning on very large data sets." In Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on, vol. 3, pp. 2579-2584. IEEE, 1998.

[7] Gong, Shucheng, and Hongyan Liu. "Constructing Decision Trees for Unstructured Data." In Advanced Data Mining and Applications, pp. 475-487. Springer International Publishing, 2014.

[8] Imam, Tasadduq, Kai Ming Ting, and Joarder Kamruzzaman. "z-SVM: an SVM for improved classification of imbalanced data." In AI 2006: Advances in Artificial Intelligence, pp. 264-273. Springer Berlin Heidelberg, 2006.

[9] Cao, Peng, Dazhe Zhao, and Osmar Zaiane. "An optimized cost-sensitive SVM for imbalanced data learning." In Advances in Knowledge Discovery and Data Mining, pp. 280-292. Springer Berlin Heidelberg, 2013.

[10] Tang, Yuchun, Yan-Qing Zhang, Nitesh V. Chawla, and Sven Krasser. "SVMs modeling for highly imbalanced classification." Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 39, no. 1 (2009): 281-288.

[11] Chawla, Nitesh V., Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. "SMOTE: synthetic minority over-sampling technique." Journal of artificial intelligence research (2002): 321-357.

[12] Mani, Inderjeet, 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.

[13] Brodersen, Kay H., Cheng Soon Ong, Klaas E. Stephan, and Joachim M. Buhmann. "The balanced accuracy and its posterior distribution." In Pattern Recognition (ICPR), 2010 20th International Conference on, pp. 3121-3124. IEEE, 2010.

[14] http://www.gartner.com/technology/research/methodologies/hype-cycle.jsp

[15] Tan, Songbo. "Neighbor-weighted k-nearest neighbor for unbalanced text corpus." *Expert Systems with Applications* 28.4 (2005): 667-671.

[16] Yanmin Sun, Mohamed S. Kamel, Andrew K.C. Wong, Yang Wang, Cost-sensitive boosting for classification of imbalanced data, Pattern Recognition, Volume 40, Issue 12, December 2007, Pages 3358-3378, ISSN 0031-3203, http://dx.doi.org/10.1016/j.patcog.2007.04.009