Cost Sensitive Class Imbalance Learning using ANFIS

Mr M Satya Srinivas, Dr A Yesubabu, Dr G Pradeepin

Research Scholar at K L University, Tadepalligudem
Prof. & HOD CSE Dept, CRR College of Engineering
Assoc. Prof. CSE Dept, K L University

Address For Correspondence:
Mr M Satya Srinivas, Research Scholar at K L University, Tadepalligudem
E-mail: maddipativas@gmail.com

ABSTRACT

In many real world problems the data to be classified may be imbalanced which might be a major problem for effective classification. Due to imbalance nature of Data, the classifier was biased towards majority class and hence minority class samples might be misclassified. For example In Intruder Detection systems, the strength of non intrusive class is high compared to intrusive. So the constructed classifier was biased towards non intrusive. Due to this biasing the intrusive samples may be misclassified as non intrusive.

There are various methods for balancing the dataset. Over sampling balances the dataset by generating new samples in minority class. Under sampling balances the dataset by eliminating samples in majority class until the strengths of both classes are approximated.

1.2 Cost Sensitive Problem:

In real world datasets, the classes may have cost differentiation, which means cost of misclassifying a sample form different classes may be different. For example, In Intruder detection systems, the misclassifying cost of intrusive is high compared to non intrusive. In such applications, the performance of classifier will be judged by total misclassification cost rather than error rate.

The misclassification cost of misclassifying sample ‘x’ can be determined by using formulae

\[ L(x, i) = \sum_j P(j|x)C_{ij} \]  

INTRODUCTION

Class Imbalance Problem:

From decades of years Data mining plays a major role in providing solutions for real world problems. In some real world problems such as Oil spill detection, Fraud detection and Network intruder detection systems the data to be classified was imbalanced. Due to Imbalance nature of the data the constructed classifier may be biased towards majority class and hence minority class samples might be misclassified. For example In Intruder Detection systems, the strength of non intrusive class is high compared to intrusive. So the constructed classifier was biased towards non intrusive. Due to this biasing the intrusive samples may be misclassified as non intrusive.

There are various methods for balancing the dataset. Over sampling balances the dataset by generating new samples in minority class. Under sampling balances the dataset by eliminating samples in majority class until the strengths of both classes are approximated.
Where $P(j|x)$ is probability of classing a sample belong to class ‘x’ as class ‘j’
$C_{ij}$ is cost of misclassifying sample ‘i’ as ‘j’

In this paper, Section III discuss about various algorithms for class Imbalance and cost sensitive problems in classification. Section IV discuss about

### 2 Related Work:

Many methods have been proposed for balancing imbalanced data by considering experimentation on various data set. A survey on methods for learning from imbalanced dataset (Haibo He, 2010). To address cost sensitive class imbalance problem, Zhi-Hua Zhou and Xu-Ying Liu introduced neural networks method (Zhi-Hua Zhou 2006). Support vectors machines with granular computing can used for training highly imbalanced dataset, this can be found in (Y Tang 2010). Yanmin Suna proposed boosting methods for cost sensitive classification of imbalanced data (Y suna, 2007). A lot progressing was carried in network intruder detection with data mining, Qinglei Zhang proposed Support Vector machines for classifications of intruder detection dataset (Q Zhang 2007). Yang Yi extended the research by using incremental SVM for classification of IDS dataset (Yang Yi 2011). In FSVM method different membership values are assigned to different example to reflect their importance (R Batuvita 2010).

### 3. Background:

#### 3.1 Adaboost:

Adaboost algorithm works by

1. constructing weak learners $h_i: X \to [0,1]$ ‘X’ is set of inputs and [0,1] is the desired output
2. Find a weak learner that minimizes total weighted error $E_t = \sum W_i e^{-y_i h(x_i)}$
3. Choose $\alpha_t = \ln \{ \frac{1 - E_t}{E_t} \} / 2$ (2)
4. Add to ensemble
5. Update weight vector
6. Renormalize the weights.

#### 3.2 CSSVM:

SVM constructs a classifier by maximizing margin in a hyper plane which separates classes. However, it is overwhelmed by the majority class instances in the case of imbalanced datasets because the objective of regular SVM is to maximize the accuracy. Cost Sensitive SVM assigns variable costs to support vectors. The hyper plane will be shifted towards high cost support vectors, such that the probability of misclassification of high cost support vectors will be reduced and hence decreases misclassification rate (Y Tang, 2010).

#### 3.3 GSVM(Granular based SVM):

GSVM follows Divide & conquer strategy. GSVM works by grouping majority class samples and minority class samples into granules. The granules are separated by hyper plane. The advantage of GSVM is the efficiency of separation will be improved if they are separated as granules(Y Tang 2010).

#### 3.4 FSVM(Fuzzy SVM):

In FSVM method different examples are assigned with different membership values which reflects their importance[8]. More important examples are assigned with high membership values while less important examples are assigned with low membership values such that SVM soft margin optimization problem is formulated as

$$\min \left( \frac{1}{2} W^* W + C^* \sum_{i=1}^{l} m_i \varepsilon_i \right) \quad \text{s.t.} \quad y_i (W^* \Phi(x_i) + b) \geq 1 - \varepsilon_i \quad \text{and} \quad \varepsilon_i > 0$$ (3)

#### 3.5 Cost Sensitive Neural Networks:

Most of the neural network classifiers are constructed using multi layer perceptron model with back propagation algorithm in tandem with gradient descendental rule. The error function in back propagation algorithm is defined as

$$E(w) = \frac{1}{2} \sum (t_k - o_k)^2$$ (4)

Where w is the set of weights and $t_k$ and $o_k$ are the target and actual outputs. Gradient descendental rule is used to adjust the weights of the neural networks.
\[ \Delta W = -\eta \frac{d}{dW}(E(W)) \quad (5) \]

In cost sensitive neural networks the cost is included in learning rate, error computation and weight adjustments. The key ideas of this approach are 1) Decrease learning rate for costly examples 2) Include penalty factor for error computation of costly examples 3) Assign more weights for computation of weight adjustments of costly examples.

Another popularly used neural network for constructing classifier is RBFN (Radial Basis Functional Network)

RBFNs are based on interpolation and approximation theory. RBFNs are functionally equivalent to Fuzzy inference system. RBFNs are constructed by using receptive field units. The activation value of the i\textsuperscript{th} receptive field unit is defined as

\[ W_i = R_i(x) = R_i(||x-\mu_i||/\sigma_i) \quad i=1,2,\ldots,H \]

\( x \) is the input vector and \( \mu_i \) is the mean of the input vectors belonging to the same receptive field. \( H \) is the number of receptive fields. \( R_i(.) \) is the i\textsuperscript{th} receptive field function. Generally Gaussian function is used as radial basis function.

Another popularly used neural network for constructing classifier is RBFN (Radial Basis Functional Network)

RBFNs are based on interpolation and approximation theory. RBFNs are functionally equivalent to Fuzzy inference system. RBFNs are constructed by using receptive field units. The activation value of the i\textsuperscript{th} receptive field unit is defined as

\[ W_i = R_i(x) = R_i(||x-\mu_i||/\sigma_i) \quad i=1,2,\ldots,H \]

\( x \) is the input vector and \( \mu_i \) is the mean of the input vectors belonging to the same receptive field. \( H \) is the number of receptive fields. \( R_i(.) \) is the i\textsuperscript{th} receptive field function. Generally Gaussian function is used as radial basis function.

**Fig. 1: RBFN**

The output of RBFN is computed in two ways. One simple method is the weighted sum of receptive field associated with each unit.

\[ d(x) = \sum_{i=1}^{H} C_i W_i \quad (6) \]

Another method is the overall output is the weighted average of receptive field associated with each unit.

\[ d(x) = \frac{\sum_{i=1}^{H} C_i W_i}{\sum_{i=1}^{H} W_i} \quad (7) \]

### 4 Empirical Study:

We conducted experiments using Multi layer perceptrons using Back propagation (MLP BP), Radial Basis function Networks (RBFN), Adaboost, Support vector machines (SVM), Cost sensitive neural networks (CS NN), Cost sensitive SVM and Cost sensitive C45.

The average performance of cost sensitive SVM found to be satisfactory compared to remaining algorithms for most of the datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>CS_NN</th>
<th>MLP_BP</th>
<th>SVM</th>
<th>Ada Boost</th>
<th>CS_SVM</th>
<th>CS_C45</th>
<th>RBFN</th>
</tr>
</thead>
<tbody>
<tr>
<td>wisconsin</td>
<td>0.77086</td>
<td>0.96236</td>
<td>0.96657</td>
<td>0.9707</td>
<td>0.96363</td>
<td>0.95981</td>
<td></td>
</tr>
<tr>
<td>iris</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.99</td>
<td>1</td>
<td>0.99</td>
<td>0.99</td>
</tr>
<tr>
<td>new-thyroid1</td>
<td>0.91944</td>
<td>0.85992</td>
<td>0.98293</td>
<td>0.9686507</td>
<td>0.98293</td>
<td>0.97460</td>
<td>0.94881</td>
</tr>
<tr>
<td>glass1</td>
<td>0.60694</td>
<td>0.58716</td>
<td>0.49629</td>
<td>0.7875264</td>
<td>0.62640</td>
<td>0.71600</td>
<td>0.77901</td>
</tr>
<tr>
<td>haberman</td>
<td>0.57646</td>
<td>0.62446</td>
<td>0.50361</td>
<td>0.5815277</td>
<td>0.61969</td>
<td>0.57517</td>
<td>0.58779</td>
</tr>
<tr>
<td>segment0</td>
<td>0.5</td>
<td>0.50202</td>
<td>0.99267</td>
<td>0.9916433</td>
<td>0.99646</td>
<td>0.99189</td>
<td>0.77409</td>
</tr>
</tbody>
</table>

### 5 ANFIS:

#### 5.1 ANFIS (Advanced Neuro Fuzzy Inference system):
Adaptive neuro fuzzy inference system (ANFIS) is a kind of neural network that is based on TSK fuzzy inference system. Since it integrates both neural networks and fuzzy logic principles, it has potential to capture the benefits of both in a single framework.

ANFIS architecture is functionally equivalent to fuzzy inference system for sugeno fuzzy model. It is a five layered architecture.

Layer 1: Every node i in this layer is an adaptive node with node function
\[ O_{1,i} = \mu_{A_1}(x) \text{ for } i=1,2 \text{ or } O_{1,i} = \mu_{B_1}(y) \text{ for } i=3,4 \] (8)

Layer 2: Every node in this layer is a fixed node with output as the product of all the input signals. The membership function uses premises parameters.
\[ O_{2,i} = \mu_{A_1}(x) * \mu_{B_1}(y) = W_i \text{ for } i=1,2 \] (9)

Layer 3: Every node in this layer is a fixed node with output as the firing strength of fuzzy rule.
\[ O_{3,i} = \frac{W_i}{W_1 + W_2} \text{ for } i=1,2 \] (10)

Layer 4: Every node in this layer is an adaptive node with a node function
\[ O_{4,i} = W_i \text{ for } i=1,2 \] (11)

Layer 5: The single node in this layer computes overall output as the sum of all incoming signals.
\[ O_{5,i} = \sum_{i} W_i f_i \] (12)

5.2 Cost sensitive ANFIS:
Cost sensitivity in class imbalance problem can be solved by using ANFIS architecture.
In ANFIS architecture Layer 1 and layer 4 are the adaptive layers with premise and consequent parameters respectively.

There are two approaches to apply cost sensitivity to ANFIS architecture.
1. Apply cost sensitivity to consequent parameters
2. Apply cost sensitivity to premises parameters

5.2.1 Apply cost sensitivity to consequent parameters:
Consequent parameters are identified by method of least squares
\[ O = (A^T A)^{-1} A^T Y \] (13)
Where ‘O’ is the set of consequent parameters(p,q,r).
‘A’ is the matrix of coefficients of variables in training set
‘Y’ is the set of target values.
The error in least square estimator can be calculated by
\[ E(O) = (Y - A O)^T W (Y - A O) \] (14)
By applying cost sensitivity, the weight(cost) vector of the training samples is included in computation of error function and hence the error function is
\[ E_w(O) = (Y - A O)^T W W (Y - A O) \] (15)

5.2.2 Apply cost sensitivity to premises parameters:
Premise parameters are updated by back propagation algorithm with gradient descendent.
\[ \Delta W_n = -\eta \frac{d}{d_w} (E(W_n)) \]  

Cost sensitivity can be applied by including cost vector to adjust the weight vector in multilayer perceptron and hence the weight adjustment becomes

\[ \Delta W_n = -\eta^* C^* \frac{d}{d_w} (E(W_n)) \]  

\( H \) is learning rate and ‘C’ cost vector and \( W \) is weight vector.

**Experiments & Results:**

We considered Software defect prediction dataset, the performance of the classifier is determined by precision, recall, F-measure and G-mean.

\[
\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}
\]

\[
\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}
\]

We conducted the experiments by using R and Rattle software, the performance of various algorithms adaboost, KSVM, ANOVA SVM, Neural Networks and ANFIS are assessed and the results of ANFIS are found satisfactory compared with Precision and recall values to ada boost, KSVM, ANOVA SVM and Neural networks using SDP dataset.

**Conclusion:**

Cost sensitivity class imbalance problem is solved by various algorithms like Ada boost, CS SVM, FSVM and Neural networks. The performance of these algorithms are assessed. The performance of classifier can be improved by using Adaptive Neuro Fuzzy Inerence systems. In ANFIS-CIL cost sensitivity is applied to premises and consequent parameters of ANFIS. Use of Cost sensitive ANFIS might reduce misclassification cost of Cost sensitive imbalanced datasets.
REFERENCES


https://github.com/KarloKnezevic/ANFIS.


