

A Methodology for Optimizing the Cost Matrix in Cost Sensitive Learning Models applied to Prediction of Molecular Functions in Embryophyta Plants

S. García-López¹, J. A. Jaramillo-Garzón^{1,2}, L. Duque-Muñoz^{1,2} and C. G. Castellanos-Domínguez¹

¹Signal Processing and Recognition Group, Universidad Nacional de Colombia,
Campus la Nubia, Km 7 vía al Magdalena, Manizales, Colombia

²Grupo de Máquinas Inteligentes y Reconocimiento de Patrones - MIRP, Instituto Tecnológico Metropolitano,
Cll 54A No 30-01, Medellín, Colombia

Keywords: Molecular Functions Prediction, Proteins, Cuckoo Search, Cost Sensitive Learning, Class Imbalance.

Abstract: Due to the large amount of data generated by genomics and proteomics research, the use of computational methods has been a great support tool for this purpose. However, tools based on machine learning, face several problems associated to the nature of the data, one of them is the class-imbalance problem. Several balancing techniques exist to obtain an improvement in prediction performance, such as boosting and resampling, but they have multiple weaknesses in difficult data spaces. On the other hand, cost sensitive learning is an alternative solution, yet, the obtention of appropriate cost matrix to induce a good prediction model is complex, and still remains an open problem. In this paper, a methodology to obtain an optimal cost matrix to train models based on cost sensitive learning is proposed. The results show that cost sensitive learning with a proper cost can be very competitive, and even outperform many class-balance strategies in the state of the art. Tests were applied to prediction of molecular functions in Embryophyta plants.

1 INTRODUCTION

Modern biology has seen an increasing use of computational techniques for large scale and complex biological data analysis. Various computational techniques, particularly machine learning algorithms (Larrañaga et al., 2006) are applied to identify functions of gene products specified by the molecular activities they perform. In this context, there is a vast number of problems associated with the nature of the data. In particular, given that the same protein can be associated to several functional classes, a problem of classification with multiple labels is generated. A straightforward way to solve this kind of problem is the “one-against-all” strategy, in which a binary classifier is trained per each class, in order to take independent decisions about the membership of proteins. Yet, this approach leads to a high degree of imbalance between the number of samples in each class, magnifying the already present disparity in their sizes and thereby producing a large bias towards the category with more information (Sonnenburg et al., 2007). There are several ways to address class imbalance problems. Techniques like Sampling and Boosting offer different so-

lutions to same issue, either from the addition of subtraction of samples to balance class distribution (He and Garcia, 2009), or by the training of individually trained classifiers in an iterative way in order to emphasize on the incorrectly learned instances by the previous iteration trained classifier (Ding, 2011). However, these techniques have some drawbacks, between them, the over-training and noise addition in the training set (oversampling), the loss of useful data if a reliable sample selection criteria is not selected (subsampling), tendency to fail if there not exist enough data or the inability to be a good model in the presence of noise in the training set (Boosting) (He and Garcia, 2009). By the other hand, models based on cost sensitive learning assume different costs (or penalties) when examples are misclassified from one category to another. This process is modelled by a cost matrix that is a numerical representation of the penalty of classifying examples from one category to another. Conventionally, models based on cost-sensitive learning assume that the costs are fixed, but this condition is not met in real-world applications and this is still an open problem (Liu and Zhou, 2012).

In this paper, a simple and efficient methodology

for obtaining the optimal cost matrix of a cost sensitive learning model is proposed. This methodology is applied to the prediction of molecular functions in Embryophyta plants and is compared with a broad spectrum of class-balance strategies in order to obtain a comprehensive analysis of the problem. The results show that cost sensitive models are highly reliable and can outperform many commonly used balance strategies in the prediction of molecular functions.

2 CLASS-BALANCE STRATEGIES

This section describes the principles of all the commonly used class-balance strategies that are going to be used in the experiments below. They are divided into three categories: sampling strategies, boosting strategies and cost sensitive strategies.

2.1 Sampling Strategies

2.1.1 Synthetic Minority Oversampling Method (SMOTE)

SMOTE is an oversampling method proposed in (Chawla et al., 2002), which main idea is to create new synthetic samples that will belong to the minority class. These samples are computed by interpolation among several closely spaced real samples. In this way, the decision boundary of the minority class becomes more general (Grzymala-Busse et al., 2005). The synthetic samples are generated as follows: for each real sample under consideration, represented as a feature vector, the distance between it and its nearest neighbors is taken. The result is multiplied by a random number between 0 to 1 with a uniform probability, and this result is added to the original feature vector. This procedure causes the selection of a random point along the line segment between two samples. The SMOTE algorithm can be seen in (Chawla et al., 2002).

2.1.2 Subsampling based on Particle Swarm Optimization

This technique is based on the search of an optimal sample subset for majority class, that maximizes the generalization capability of the classifier. To this purpose, a metaheuristic optimization strategy known as Particle Swarm Optimization (PSO) is used. The main concept of this technique is synthesized in the following form: To a given dataset, a cross validation composed by three folds is used, generating ex-

ternal training and test sets. This external training sets are partitioned in turn with another cross-validation of three folds, these being the internal validation sets. In this internal validation set, the internal training sets are used to resampling, while internal test sets guide the optimization process. Then the external test sets are reserved to the balance set evaluation and excluded from resample process. The hybrid optimization system based on PSO is used to evaluate the merit of each majority class sample to compensate the balance effect between them. This is achieved through the creation of different samples subsets of majority class combined with the minority class to build a classification model which is them used to the partition of test classification. When the completion criterion is accomplished, the selected samples by the last iteration are ordered by their frequency selection. After the list of frequencies of selected samples is obtained, a balanced dataset from the combination of the samples belonging to majority class with major frequency index and samples of minority class is constructed (Yang et al., 2009). the process of this algorithm is explained in greater detail in (Yang et al., 2009).

2.2 Boosting Strategies

2.2.1 AdaBoost

Boosting algorithms are iterative algorithms that place different weights on the training distribution at each iteration. After each iteration, boosting increases the weights associated with incorrectly classified examples and decreases the weights associated with correctly classified examples. This forces the system to focus on the rare items, incrementing the weights assigned to rare classes. The most representative technique belonging to Boosting algorithms is AdaBoost (Polikar, 2006). AdaBoost generates a set of classifiers, and combines them through weighted majority voting of the classes predicted by the individual hypotheses. The hypotheses are generated by training a weak classifier, using instances drawn from an iteratively updated distribution of the training data. This distribution update ensures that instances misclassified by the previous classifier are more likely to be included in the training data of the next classifier. Hence, consecutive classifiers training data are geared towards increasingly hard-to-classify instances (Polikar, 2006), (Schapire, 1999). AdaBoost takes the final decision via weighted majority voting, i.e, each classifier will have a different power of decision, it depends on the performance during training procedure, as the classifier has better performance, will be favored with a greater power of decision over the other

classifiers (Polikar, 2006). The algorithm is explained in a detailed way in (Schapire, 1999).

2.3 Cost Sensitive Strategies

2.3.1 Cost Sensitive Learning

This strategy attempts to minimize costs (or maximize profits) associated with its decisions rather than simply getting a high precision. In biclass problems, when misclassified samples of one class are much more costly than misclassified samples of another class, it generate a model that center more in the correct classification of sample of the most costly class samples than a model where the class treated equally. Given a costs specification for correct and incorrect predictions, a sample could be predicted to have the class that leads to lower expected cost, where the expected value is computed using conditional probability of each class given a sample. Mathematically be (i, j) the inputs associated to a cost matrix C , where C have the cost to predict a class i when the true class is j . If $i = j$, then the prediction is correct, while if $i \neq j$, the prediction is incorrect. The optimal prediction for a sample is the class that minimize:

$$L(x, i) = \sum_j P(j|x)C(i, j)$$

Where cost matrix $C(i, j)$ is defined as:

Table 1: Cost matrix.

| | Actual negative | Actual positive |
|------------------|--------------------|--------------------|
| Predict negative | $C(0, 0) = c_{00}$ | $C(0, 1) = c_{01}$ |
| Predict positive | $C(1, 0) = c_{10}$ | $C(1, 1) = c_{11}$ |

For each i , $L(x, i)$ is the sum over the alternative possibilities for the true class of x . In this framework, the goal of algorithm based on cost sensitive learning is to produce a classifier that can estimate the probability $P(j|x)$ given any example x , being this the true class of x . For an example x , i means make the prediction act as if i was the true class of x . the essence decision-making by cost sensitivity is that this may be optimal to act as if a class is true even when other classes are more likely (Elkan, 2001). In biclass case, the optimal prediction will be the class 1 if and only if the expected cost of the prediction is less than or equal to the expected cost of predicting class 0, i.e:

$$P(j=0|x)c_{10} + P(j=1|x)c_{11} < P(j=0|x)c_{00} + P(j=1|x)c_{01}$$

Which equals to:

$$(1-p)c_{10} + pc_{11} < (1-p)c_{00} + pc_{01}$$

where $p = P(j=1|x)$

The threshold for optimal decision making is such that:

$$p^* = \frac{c_{10} - c_{00}}{c_{10} - c_{00} + c_{01} - c_{11}}$$

2.3.2 MetaCost

The basic idea of MetaCost is to take a normal, unaltered classifier and adjust the learning with a cost matrix. First, the training set is taken to form multiples subsets via bootstrap. each subset create by bootstrap is used to build a ensemble of classifiers to take the final decision, where each subsets and classifiers are equals to number of iteration used in MetaCost. The ensemble of classifiers are then combined through a majority vote to determine the probability of each data object x belonging to each class label. Next, each data object in the training data is relabeled based on the evaluation of a conditional risk function, and a final classifier is then produced after applying the classification algorithm to the relabeled training data. The conditional risk function is defined as:

$$R(i|x) = \sum_j P(j|x)C_{i,j} \quad (1)$$

Where the conditional risk determine the cost of predicting that sample x belongs to class label i instead of class label j , $P(j|x)$ is the conditional probability that sample x belongs to label j , and $C_{i,j}$ is the cost matrix used in the classification (Domingos, 1999). The Algorithm is explained in detail in (Domingos, 1999)

3 PROPOSED METHOD: OPTIMAL COST MATRIX SEARCH VIA CUCKOOCOST

This section describes the theoretical background and foundations of the proposed methodology for selecting the optimal cost matrix.

3.1 Cuckoo Search

Cuckoo Search is based on the parasitic behavior exposed by some species of Cuckoo birds. His natural strategy consist in leave eggs in host nest created by other birds. This eggs presents the particularity to

have a big similitude with host eggs, the more similar they are, the greater your chance of survival.

Based on this statement, Cuckoo Search use three idealized rules:

- Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest.
- The best nests with high quality of eggs (solutions) will carry over to the next generations.
- The number of available host nests is fixed, and a host can discover an alien egg with a probability $p_a \in [0, 1]$. In this case, the host bird can either throw the egg away or abandon the nest so as to build a completely new nest in a new location.

For simplicity, this last assumption can be approximated by a fraction P_a of the n nests being replaced by new nests (with new random solutions at new locations). The generation of new solutions is defined as:

$$x_i^{(t+1)} = x_i^t + \alpha \oplus Levy(\lambda) \quad (2)$$

Being λ the step size and Lévy flights provides a random walk to move around the search space. The Lévy flight can be expressed as:

$$x_i^{(t+1)} = x_i^t + \alpha \oplus Lvy(\lambda) \quad (3)$$

Where

$$Levy \sim u = t^{-\lambda}, (1 < \lambda < 3) \quad (4)$$

3.2 CuckooCost

In biclass problems, the category with the most lower representation or among of samples has a higher misclassification cost C^+ (usually this samples corresponds to category of interest or minority class). Moreover, the category with more samples have a lower misclassification cost C^- , due to big amount of data, helping to its representation. Taking in account this fact, if a cost matrix is given, the decision that are optimal are unchanged if their cost (in this case the inputs of matrix cost) is multiplied by a scalling factor (Liu and Zhou, 2006), this normalization allow change of baseline in which cost are measured. Therefore, if each elements of cost matrix is multiplied by $\frac{1}{C^-}$, it can be expressed as:

Table 2.

| | Actual negative | Actual positive |
|------------------|-----------------|--------------------|
| Predict negative | $C(0,0) = 0$ | $C(0,1) = C^+/C^-$ |
| Predict positive | $C(1,0) = 1$ | $C(1,1) = 0$ |

Since costs can be normalized with the optimal decision unchanged, C^- can always be set to 1, and

therefore C^+/C^- is always bigger than 1 (Elkan, 2001), this relations is called called cost-sensitive rescale ratio or cost ratio (Liu and Zhou, 2006). In order to deal with class-imbalance using Rescaling, different costs are to be incurred for different classes. So, the optimal rescale ratio (called imbalance rescale ratio) of positive class to negative class $ri_{+,-}$ is defined a:

$$ri_{+,-} = N^-/N^+$$

So to handle unequa misclassification and class-imbalance at the same time, both the cost-sensitive rescale ratio rc and the imbalance rescale ratio ri should be take in consideration (Liu and Zhou, 2006). Merging scale factors, we can obtain:

$$\phi = rc * ri_{+,-}$$

Being ϕ the cost ratio of matrix cost, where $\phi \geq ri_{+,-}$. CuckooCost use Cuckoo Search to obtain the optimal parameter values to achieve the best classification performance possible. each nests represents a set of solutions in the search space, i.e, each egg on the nest represent a parameter that will be used in the model optimization, in this case the cost ratio and classifier parameters to improve the performance of cost sensitive learning. In Algorithm 1 explain in detail CuckooCost. It is important notify that in Cuckoo Search, the parameters P_a and α help to explore efficiently the search space and allow to find globally and locally improved solutions, respectively. Additionally, these parameters directly influence the convergence rate of optimization algorithm, for instance, if value of P_a tends to be small and α value is large, the performance of the algorithm will be poor, which induce a increment in number of iterations to converge into a optimal value. if on the contrary, the value of P_a is large and value of α is small, the speed of convergence, the convergence speed of the algorithm tends to be very high to obtain the best solution (Valian et al., 2011). Usually, both α and P_a use fixed values, this may augment the probability to decrease the efficiency of the algorithm. To avoid this problem, a improvement to Cuckoo Search proposed in (Valian et al., 2011) is used, which consist in use a range of P_a and α to change dynamically in each iteration this values, through the following equations:

$$c = \frac{1}{N_{tot}} \ln \left(\frac{\alpha_{min}}{\alpha_{max}} \right) \quad (5)$$

$$P_a = P_{max} - \frac{N_{iter}}{N_{tot}} (P_{max} - P_{min}) \quad (6)$$

$$\alpha = \alpha_{max} \exp(cN_{iter}) \quad (7)$$

Table 3: Dataset definition.

| Ontology | Class | Biological name | Samples | Imbalance ratio |
|--------------------|------------|---|---------|-----------------|
| Molecular Function | GO 0003677 | DNA binding | 143 | 1 : 7.68 |
| | GO 0003700 | Sequence-specific DNA binding transcription factor activity | 102 | 1 : 10.76 |
| | GO 0003824 | Catalytic activity | 401 | 1 : 2.74 |
| | GO 0005215 | Transporter activity | 133 | 1 : 8.26 |
| | GO 0016787 | Hydrolase activity | 237 | 1 : 4.63 |
| | GO 0030234 | Enzyme regulator activity | 46 | 1 : 23.87 |
| | GO 0030528 | Transcription regulator activity | 152 | 1 : 7.22 |

The best nest will contain the optimal parameters to induce a dependable cost sensitive model.

4 EXPERIMENTAL SETUP

4.1 Database

The database is constituted by 1098 proteins belonging to Embryophyta taxonomy of the Uniprot database (Jain et al., 2009) with at least one annotation in the molecular function ontology of the Gene Ontology Annotation project (Ashburner et al., 2000). Sequences predicted by computational tools and with no real experimental evidence were discarded. Proteins are associated to one or more of the seven categories shown in Table 3. The dataset does not contain protein sequences with a sequence identity superior to 40% in order to avoid bias and overtraining in the training dataset.

4.2 Characterization of Protein Sequences

All the proteins (input space) were mapped into feature space. This set of features is composed by three groups of attributes: physical-chemical features, primary structure composition statistics and secondary structure composition statistics (see Table 4).

The first group reveals information about the biochemical properties of the molecules, and it is composed by: molecular weight, polarity of amino acid side chains, isoelectric point, and hydropaticity index (GRAVY). In the second group, the frequencies of each amino acid and the frequencies of all possible n-grams of fixed length n was extracted, where $n = 1, 2$. Subsequently, in the last set, an estimate of the secondary structure of each protein, using the Predator software 2.1 was made (Frishman et al., 1997), such as the percentage of each structure (alpha, beta, coiled coils) and each "di-gram" (9 in total, representing the

Algorithm 1: CuckooCost algorithm.

Require: P_a and ranges of P_a values: $P_{a_{min}}, P_{a_{max}}$

Require: α and range of α values: $\alpha_{min}, \alpha_{max}$

Require: Number of nest: $NumberNest$

Require: Number of eggs per nest: $eggdimension$

Require: Total number of iterations: N_{tot}

Require: location of best nest: ind

Require: local best nest: $LBest$

// set up the initial nests randomly and initial set of values belonging to fitness function

$N_{iter} \leftarrow 0, P_a \leftarrow initval, \alpha \leftarrow initval2$

$Cuset \leftarrow initNests(eggdimension)$

$ftset \leftarrow \mathbf{O}(nullvector)$

//Obtain the initial best solution from initial nests

$(ftset, Best, ind) \leftarrow getBest(Cuset, Cuset, ftset)$

$c \leftarrow \frac{1}{N_{tot}} \ln \left(\frac{\alpha_{min}}{\alpha_{max}} \right)$

while $N_{iter} < N_{tot}$ **do**

$P_a = P_{max} - \frac{N_{iter}}{N_{tot}} (P_{max} - P_{min})$

$\alpha = \alpha_{max} \exp(cN_{iter})$

//Generate new solutions, but keep the current best

$neoNests \leftarrow getCuckoos(Cuset, Best, \alpha)$

$(fnew, LBest, ind) \leftarrow getBest(Cuset, neoNest, ftset)$

$N_{iter} = N_{iter} + NumberNest$

//Discovery and randomization

$discover \leftarrow EmptyNests(Cuset, P_a, maxindex)$

$(fnew, LBest, ind) \leftarrow getBest(Cuset, discover, ftset)$

$N_{iter} \leftarrow N_{iter} + NumberNest$

//Find the best objective so far

if $fnew > fmax$ **then**

$fmax \leftarrow fnew$

$Best \leftarrow LBest$

end if

end while

return $Best$

Table 4: Description of feature space.

| Feature | Description | Number |
|---------------------|---|------------|
| Chemical-Physical | Length of the sequences | 1 |
| | Molecular weight | 1 |
| | Percentage of positively charged residues (%) | 1 |
| | Percentage of negatively charged residues (%) | 1 |
| | Isoelectric point | 1 |
| Primary Structure | GRAVY - Hydropathic index | 1 |
| | Frequency of each aminoacids | 20 |
| Secondary Structure | Frequency of each dimers | 400 |
| | Frequency of structures | 3 |
| | Frequency of dimers in structures | 9 |
| TOTAL | | 438 |

combinations of alpha, beta and coiled coils) were extracted. The estimation of the secondary structure of the proteins was made from the data based on the primary structure. Thus, none of the secondary structures reported here were calculated from known data. The total set contains 438 feature attributes.

4.3 Feature Selection

In order to obtain representative characteristics, the feature selection was performed as a pre-processing stage from the relevance and redundancy analysis. The relevant characteristics were quantified by calculating the correlation with the actual labels for all features. The redundant features were identified through the analysis of the feature correlation matrix of dimension $n \times n$. To reduce computational cost, a fast filter-selection algorithm proposed in (Yu and Liu, 2004) was used. As a selection criterion, a measure based on non-linear correlation was used.

4.4 Class Imbalance and Classification Schemes

To mitigate the effect generated by multi-label samples in the dataset, reduce classification complexity and to obtain a better interpretation of results, a against vs all learning strategy was used. Nevertheless, the use of this strategy raises in additional problems such as highly class imbalance in the data space. To overcome the unbalanced data, five class balance strategies are applied. Between these techniques, are: AdaBoost (Ada)(Schapire, 1999), SMOTE (Chawla et al., 2002), Subsampling based on particle swarm optimization (SPSO) (Yang et al., 2009), cost sensitive learning (CS)(Elkan, 2001) and MetaCost (MC)(Domingos, 1999) without matrix cost optimization via CuckooCost (CS)(MC), and cost sensitive learning and MetaCost within CuckooCost (CSCu),(MCCu). To all classification tests,

support vector machines (SVM) with Gaussian Kernel was used, except the test with AdaBoost. In this case, it was necessary the use of Naive Bayes as weak classifier and twenty iterations for Boosting technique.

The tuning of parameters presents in SVM and Gaussian Kernel (penalty constant C and dispersion γ) were made with particle swarm optimization (PSO). Taking as objective function the maximization of adjustable geometric mean (AGM) (Batuwita and Palade, 2009), which have the property to improve the sensitivity, keeping reduction of specificity at minimal. Noteworthy that PSO was not used in cost sensitive learning strategies (CS and MetaCost), due to two reasons: i) initially the methodology was proposed based on PSO, however, by not getting good results, the method was adapted with optimization based on Cuckoo Search, ii) CuckooCost take γ and penalty constant C as hyperparameters in the optimization problem. To evaluate the performance of molecular function classification, a cross-validation with ten folds was used. For CuckooCost, the search range to each parameter are:

$$\begin{aligned}
 1 &\leq \phi \leq 1.5R_d \\
 0.00030518 &\leq C \leq 4096 \\
 0.000030518 &\leq \gamma \leq 32
 \end{aligned}$$

Where ϕ is the cost ratio extracted from cost matrix, and ϕ is the imbalance ratio. Table 4 shows the different classes used on this study with its imbalance ratio and the number of samples for each class.

4.5 Evaluation Metrics

4.5.1 Performance Measures

Performance measures non-susceptible to unbalance data phenomena were used to obtain a reliably evaluation of the classification. Measures such as sensitivity, specificity, geometric mean and ROC area (AUC) were used to this purpose, which are defined as:

i) Sensitivity

$$Sensitivity = \frac{TP}{TP + FN} \tag{8}$$

ii) Specificity

$$Specificity = \frac{TN}{TN + FP} \tag{9}$$

ii) Geometric mean

$$Geometricmean = \sqrt{Sensitivity * Specificity} \tag{10}$$

iv) ROC area (AUC)

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} \quad (11)$$

Additionally, a metric that measure the degree of bias in the classification, known as the relative sensitivity (RS) (Su and Hsiao, 2007) will be used. It is defined as:

$$RS = \frac{Sensitivity}{Specificity} \quad (12)$$

4.5.2 Data Complexity Measures

The grade of data imbalance is not the only factor that leads to a biased learning. elements associated with data complexity can generate deficiencies in the learning models. Data complexity can be observed in phenomena such as difficulties inherent in data, shortcomings in classification algorithms and the low representation present in the data space (He and Garcia, 2009). The following measures were used to quantify the complexity present in data:

i) **Overlap Measures:** They examine the range and distribution of values in each category, and verify the overlap between them (Basu, 2006). In the experiment, the measures used were:

– **Volume of Overlap Region (VOR):** It measures the amount of overlap in the boundary region between two categories (Basu, 2006), and it is defined as:

$$VOR = \prod_i \frac{MIN(\max(f_i, c_1), \max(f_i, c_2)) - MAX(\min(f_i, c_1), \min(f_i, c_2))}{MAX(\max(f_i, c_1), \max(f_i, c_2)) - MIN(\min(f_i, c_1), \min(f_i, c_2))} \quad (13)$$

– **Fisher's Discriminant Ratio:** For a multidimensional problem, not necessarily all features have to contribute to class discrimination. As long as there exists one discriminating feature, the problem is easy. Therefore, we use the maximum f over all the feature dimensions to describe a problem (Basu, 2006). This measure also serves as indicator of quality in the dataset representation, i.e, if its value tends to be low, there is little contribution in the overall discrimination of the dataset, which may indicate a weak representation of the data. The Fisher's discriminant ratio is defined as:

$$Fisher = \max \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2} \quad (14)$$

– **Difference between Inter/Intra Classes Scatter Matrix:** It measures the distance between

the class distribution, The measure indicates favorability as its value is greater (García-López et al., 2012). This metric is complementary with **VOR**, **Fisher**, Fisher discriminant ratio is described as:

$$J_4 = Tr\{S_b - S_w\} \quad (15)$$

Where,

$$S_w = \sum_{i=1}^C \frac{n_i}{n} \hat{\Sigma}_i \quad (16)$$

$$S_b = \sum_{i=1}^C \frac{n_i}{n} (m_i - m)(m_i - m)^T \quad (17)$$

Being $\hat{\Sigma}_i$ the covariance matrix of i -th class, m_i the sample mean of the i -th class and m the sample mean of the whole dataset.

ii) **Measures of Geometry, Topology and Density of Manifolds:** This metrics gives indirect information about separation between categories. It is assumed that a category is composed by a collection of one or more manifolds, forming the support of the probability distribution of a given class. The shape, position and interconnectivity of manifolds gives a hint of its overlap (Basu, 2006). To evaluate the complexity of manifolds, the leave-one-out error in 1NN (LOO 1NN) is used.

5 RESULTS AND DISCUSSION

Figure 4 summarizes the results of classification that are represented by bars and lines at different color scales. Each figure contains information about the behavior of the geometric mean (red), the area under the ROC curve (AUC) (green), sensitivity (color light blue) and specificity (color light cyan). Each row depict one of the class-balance strategies, sorted in ascending order according to the strategy of balance: oversampling (**SMOTE**), subsampling (**SPSO**), cost-sensitive learning unused and using CuckooCost (**CS**, **CSCu**, **MC**, **MCCU**) and Boosting (**AdaBoost**). On the right side of the graph, it shows the dispersions of classification results obtained by each balance technique, exposed by means of boxplots.

Table 5 contains information concerning to data complexity involved in the categories. This table describes measurements that determine the overlap and separability between classes (VOR , J_4 , $Fisher$), and measurements of nonlinearity in the classifiers ($LOOerror1NN$) contrasted with information of imbalance degree for each dataset, this in order to obtain

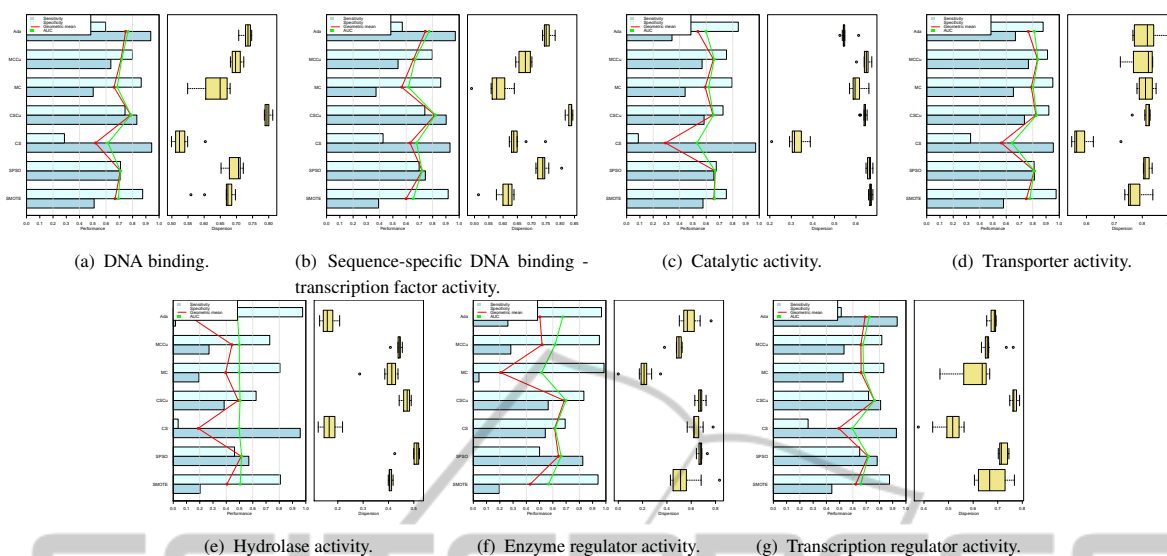


Figure 1: Molecular function prediction results.

information concerning to the difficulty to induce reliable learning models in each biclass problem. Measures such as Fisher discriminant ratio, $J4$ and tend to be favorable as they increase in value, indicating a greater separability, otherwise occurs VOR which tends to be better as its value approaches zero, indicating a smaller area of overlap. According to the values given in Table 5, The most complex space is the set belonging to Hydrolase activity (GO 0016787), showing a low value at $J4$ and VOR highest compared to other classes. This fact is proved by the results obtained for this class exhibited in Figure 1(a), Where all techniques show poor performance balance. This suggests that in this class will present a very poor representation of the data. Also, if we look again the values listed in the Table 5, The level of imbalance is not as significant as compared with the values of overlap between the data, which might lead to think that data complexity can may deteriorate more severely the learning process in protein prediction compared with the class imbalance, only when level of overlap and separability is to big compared with imbalance ratio itself. Therefore, it is convenient to use complexity measures as a complement to the level of imbalance to be certain about the difficulty of the problem.

Despite the complexity, the best behavior for Hydrolase activity was obtained SPSo, with a value of geometric mean (GM) and ROC area (AUC) just over 50% and very low dispersion in the prediction. However, the difference was very short compared to the method based on cost sensitive learning using CuckooCost (CSCu). It is remarkable that in datasets with higher imbalance between categories such as Enzyme regulator activity and Sequence-specific DNA bind-

ing transcription factor activity (GO 0030234 and GO 0003700), CSCu obtained a considerable superiority over the techniques compared, in fact, its performance overcomes in five of the seven categories (GO 0030234, GO 0003700, GO 0003677, GO 0030528 and GO 0005215), and the remaining 2 sets (GO 0016787 and GO 0003824) was one of the highest performing techniques in his prediction, as can be seen in Table 6

On the other hand, AdaBoost and SMOTE obtain the worst prediction results, especially in Hidrolase activity, Enzyme regulator activity and Transcription regulator activity (GO 0016787, 0030234 and GO 0030528). From these results we conclude that in the presence of sets with high overlap, oversampling can be conterproductive, due to there exist a high probability of adding extra noise in the training set when synthetic samples are adding, interfering with the induction of a reliable model for prediction of molecular functions. In case of AdaBoost, the high overlap can decrease considerably the generalization capability of the classifiers used by this technique, when it is forced to be rather complex decision boundaries.

An important fact shown in Figure 4 and the overall results of the Table 6, is the effect of CuckooCost in methods based on cost sensitive learning over their performance (CS,MC,CSCu,MCCu). Clearly shows a substantial improvement in MetaCost and cost sensitive learning in overall performance (increased GM and AUC), as well as the reliability of the results by decreasing dispersion in every category. Although MetaCost follows the same trend of improvement when using CuckooCost in transporter activity (GO 0005215), it is seen a slight increase in

Table 5: Table of data complexity measurements in the datasets.

| Categories | Fisher Discriminant Ratio | VOR | J4 | LOO error INN (%) | Imbalance |
|------------|---------------------------|--------------|---------|-------------------|-----------|
| GO 0003677 | 1,162564308 | 1,518414e-45 | 366,65 | 42 | 1:7,68 |
| GO 0003700 | 1,258898151 | 8,325292e-43 | 153,09 | 54,7 | 1:10,76 |
| GO 0003824 | 0,095424389 | 1,503915e-39 | 114,67 | 41,3 | 1:2,74 |
| GO 0005215 | 1,275657636 | 1,974715e-67 | 3045,37 | 19,4 | 1:8,26 |
| GO 0016787 | 0,004254501 | 6,654359e-07 | -0,472 | 53,9 | 1:4,63 |
| GO 0030234 | 0,265168845 | 1,247835e-26 | 14,712 | 79,3 | 1:23,87 |
| GO 0030528 | 0,954652410 | 1,125151e-37 | 255,43 | 37,6 | 1:7,22 |

Table 6: Table of AUC and GM.

| Categories | SMOTE | | SPSO | | CS | | CSCu | | MC | | MCCu | | Ada | |
|--------------|-------|-------|--------------|--------------|-------|-------|--------------|--------------|-------|-------|--------------|--------------|-------|-------|
| | AUC | GM | AUC | GM | AUC | GM | AUC | GM | AUC | GM | AUC | GM | AUC | GM |
| GO 0003677 | 0,693 | 0,668 | 0,708 | 0,707 | 0,615 | 0,519 | 0,788 | 0,786 | 0,684 | 0,659 | 0,718 | 0,713 | 0,766 | 0,747 |
| GO 0003700 | 0,654 | 0,599 | 0,721 | 0,721 | 0,679 | 0,629 | 0,821 | 0,817 | 0,617 | 0,566 | 0,668 | 0,655 | 0,773 | 0,744 |
| GO 0003824 | 0,664 | 0,658 | 0,667 | 0,667 | 0,53 | 0,292 | 0,655 | 0,651 | 0,618 | 0,592 | 0,661 | 0,654 | 0,599 | 0,536 |
| GO 0005215 | 0,778 | 0,752 | 0,811 | 0,81 | 0,643 | 0,562 | 0,829 | 0,823 | 0,803 | 0,788 | 0,839 | 0,835 | 0,812 | 0,766 |
| GO 0016787 | 0,505 | 0,405 | 0,516 | 0,513 | 0,497 | 0,188 | 0,504 | 0,49 | 0,499 | 0,395 | 0,499 | 0,443 | 0,485 | 0,128 |
| GO 0030234 | 0,568 | 0,429 | 0,663 | 0,642 | 0,618 | 0,613 | 0,699 | 0,686 | 0,515 | 0,205 | 0,617 | 0,518 | 0,675 | 0,502 |
| GO 0030528 | 0,659 | 0,621 | 0,717 | 0,714 | 0,595 | 0,493 | 0,763 | 0,762 | 0,68 | 0,662 | 0,676 | 0,66 | 0,723 | 0,691 |
| Total | 0,646 | 0,59 | 0,686 | 0,682 | 0,596 | 0,47 | 0,723 | 0,717 | 0,63 | 0,552 | 0,668 | 0,64 | 0,69 | 0,588 |

the variance of the result. This may be due to an appropriate number of iterations for MetaCost (10 iterations) was not taken. MetaCost use resampling via Bootstrap, taking a portion of the training set to create a subset in each iteration, then each subset is taken by a number of base classifiers equal to the number of iterations for the algorithm selected and the final classification decision is taken in committee by a vote of each classifier. When the number of iterations in MetaCost is not adequate and additionally the dataset have a substantial degree of imbalance, as it is in this case, the number of samples of interest, i.e the samples belonging to this category used for each base classifier could not be enough.

In all categories, there exist cases where some balance techniques present very similar values of GM compared with their AUC values, mainly in SPSO and CSCu. It observes that occurs particularly when the numeric difference between sensitivity and specificity is small, i.e, the numeric values of sensitivity and specificity are to close among them. This fact can be corroborated with the relative sensitivity values (RS) (Su and Hsiao, 2007), exposed in Table 7.

Table 7: Table of relative sensitivity.

| Categories | SMOTE | SPSO | CS | CSCu | MC | MCCu | Ada |
|------------|-------|-------|--------|-------|-------|-------|-------|
| GO 0003677 | 0,582 | 0,996 | 3,301 | 1,119 | 0,582 | 0,796 | 1,572 |
| GO 0003700 | 0,427 | 1,067 | 2,185 | 1,217 | 0,433 | 0,676 | 1,701 |
| GO 0003824 | 0,766 | 0,973 | 11,057 | 0,803 | 0,555 | 0,755 | 0,406 |
| GO 0005215 | 0,593 | 1,002 | 2,885 | 0,8 | 0,688 | 0,842 | 0,762 |
| GO 0016787 | 0,251 | 1,234 | 25,892 | 0,614 | 0,241 | 0,371 | 0,017 |
| GO 0030234 | 0,208 | 1,652 | 0,783 | 0,677 | 0,044 | 0,297 | 0,269 |
| GO 0030528 | 0,503 | 1,203 | 3,541 | 1,126 | 0,631 | 0,651 | 1,824 |

As it can seen, SPSO and CSCu are the techniques with less bias in their classifications, with values more close to one. The above indicates that precisely these

two classifiers try to obtain an equilibrium between sensibility and specificity values, fact that is shown with the points in Figure 4 where $AUC = GM$. Contrary to popular belief, SMOTE tends to be very specific, although sampling techniques try to become more sensitive to increase distribution of samples on category with lower representation. it is noteworthy that both CS and MC obtained a quite substantial improvement when they use CuckooCost to optimize their parameters, initially CS was to sensitive but it had a small specificity, contrary case to MC, that it had a big specificity. When CuckooCost was used, both strategies were proximal to one, specially in CS.

6 CONCLUSIONS AND FUTURE WORK

A method to optimize the free parameters associated to cost sensitive learning, applied to prediction of molecular functions in embryophyta plants was proposed, with the purpose of having direct control over sensitivity and specificity of the classification (related to the costs involved misclassifying samples belonging to each category). The optimization is proposed over the elements of the cost matrix, whose tuning was adapted on elements outside the main diagonal, building the cost ratio. The variation of the cost ratio, along with the classification parameters were used as hyperparameters in the optimization problem, since the metric intrinsically modify the fitness function. To this purpose, a metaheuristic optimization technique called Cuckoo Search was used. The methodology

takes as fitness function the maximization of adjusted geometric mean (AGM) (Batuwita and Palade, 2009). This work demonstrated that the use of models based on cost sensitivity learning are very competitive and reliable, and even superior to other balance techniques in the state of the art, specially in applications related to bioinformatics. As future work, the use of other metrics as fitness function for improving the costs associated with the classification, such as ROC area (AUC), Geometric Mean (GM), Mathews Correlation Coefficient (MCC) or another relationships between sensitivity and specificity can be considered.

ACKNOWLEDGEMENTS

This work was partially funded by the Research office (DIMA) at the Universidad Nacional de Colombia at Manizales and the Colombian National Research Centre (COLCIENCIAS) through grant No.111952128388 and the "jóvenes investigadores e innovadores - 2010 Virginia Gutierrez de Pineda" fellowship.

REFERENCES

- Ashburner, M., Ball, C., Blake, J., Botstein, D., Butler, H., Cherry, J., Davis, A., Dolinski, K., Dwight, S., Eppig, J., et al. (2000). Gene ontology: tool for the unification of biology. *Nature genetics*, 25(1):25.
- Basu, M. (2006). *Data complexity in pattern recognition*. Springer-Verlag New York Inc.
- Batuwita, R. and Palade, V. (2009). A new performance measure for class imbalance learning. application to bioinformatics problems. In *Machine Learning and Applications, 2009. ICMMLA'09. International Conference on*, pages 545–550. IEEE.
- Chawla, N., Bowyer, K., Hall, L., and Kegelmeyer, W. (2002). Smote: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research.*, 16:321–357.
- Ding, Z. (2011). *Diversified ensemble classifiers for highly imbalanced data learning and its application in bioinformatics*. PhD thesis, GEORGIA STATE UNIVERSITY.
- Domingos, P. (1999). Metacost: A general method for making classifiers cost-sensitive. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 155–164. ACM.
- Elkan, C. (2001). The foundations of cost-sensitive learning. In *International Joint Conference on Artificial Intelligence*, volume 17, pages 973–978. LAWRENCE ERLBAUM ASSOCIATES LTD.
- Frishman, D., Argos, P., et al. (1997). Seventy-five percent accuracy in protein secondary structure prediction. *Proteins-Structure Function and Genetics*, 27(3):329–335.
- García-López, S., Jaramillo-Garzón, J. A., Higuera-Vásquez, J., and Castellanos-Domínguez, C. (2012). Wrapper and filter metrics for pso-based class balance applied to protein subcellular localization. In *BIOSTEC-BIOINFORMATICS 2012*.
- Grzymala-Busse, J., Stefanowski, J., and Wilk, S. (2005). A comparison of two approaches to data mining from imbalanced data. *Journal of Intelligent Manufacturing*, 16(6):565–573.
- He, H. and Garcia, E. (2009). Learning from imbalanced data. *Knowledge and Data Engineering, IEEE Transactions on*, 21(9):1263–1284.
- Jain, E., Bairoch, A., Duvaud, S., Phan, I., Redaschi, N., Suzek, B., Martin, M., McGarvey, P., and Gasteiger, E. (2009). Infrastructure for the life sciences: design and implementation of the uniprot website. *BMC bioinformatics*, 10(1):136.
- Larrañaga, P., Calvo, B., Santana, R., Bielza, C., Galdiano, J., Inza, I., Lozano, J., Armañanzas, R., Santafé, G., Pérez, A., et al. (2006). Machine learning in bioinformatics. *Briefings in bioinformatics*, 7(1):86–112.
- Liu, X. and Zhou, Z. (2006). The influence of class imbalance on cost-sensitive learning: an empirical study. In *Data Mining, 2006. ICDM'06. Sixth International Conference on*, pages 970–974. IEEE.
- Liu, X. and Zhou, Z. (2012). Towards cost-sensitive learning for real-world applications. *New Frontiers in Applied Data Mining*, pages 494–505.
- Polikar, R. (2006). Ensemble based systems in decision making. *Circuits and Systems Magazine, IEEE*, 6(3):21–45.
- Schapire, R. (1999). A brief introduction to boosting. In *International Joint Conference on Artificial Intelligence*, volume 16, pages 1401–1406. LAWRENCE ERLBAUM ASSOCIATES LTD.
- Sonnenburg, S., Schweikert, G., Philips, P., Behr, J., and Rätsch, G. (2007). Accurate splice site prediction using support vector machines. *BMC bioinformatics*, 8(Suppl 10):S7.
- Su, C. and Hsiao, Y. (2007). An evaluation of the robustness of mts for imbalanced data. *Knowledge and Data Engineering, IEEE Transactions on*, 19(10):1321–1332.
- Valian, E., Mohanna, E., and Tavakoli, S. (2011). Improved cuckoo search algorithm for global optimization. *Int. J. Communications and Information Technology*, 1(1):31–44.
- Yang, P., Xu, L., Zhou, B., Zhang, Z., and Zomaya, A. (2009). A particle swarm based hybrid system for imbalanced medical data sampling. *BMC genomics*, 10(Suppl 3):S34.
- Yu, L. and Liu, H. (2004). Efficient feature selection via analysis of relevance and redundancy. *The Journal of Machine Learning Research*, 5:1205–1224.