ROC-BASED COST-SENSITIVE CLASSIFICATION WITH REJECT OPTION

PATTERN RECOGNITION PROJECT REPORT

PROF FRANCESCO TORTORELLA

BY
UMAMAHESWARAN RAMAN KUMAR

JUNE 18, 2017
UNIVERSITY OF CASSINO AND SOUTHERN LAZIO, ITALY

1. INTRODUCTION

The project is based on the paper 'ROC-based cost-sensitive classification with a reject option' presented in the 23rd International Conference on Pattern Recognition (ICPR). In real world scenario the cost of classification, in particular the misclassification cost, is very crucial for building a classification system. And there are cases where the classifier should reject the classification of a sample in order to avoid the huge cost of misclassification. This project implements the ROC-based ensemble method called the ROC Front method with a reject option for cost sensitive classification algorithm presented in the paper mentioned. The images shown in the report are taken from the paper specified and their reference papers mainly the 'The Multiclass ROC Front method for cost-sensitive classification'.

2. BACKGROUND

In a typical binary classification scenario, the classifier predicts the sample to be either positive (P/+1) or negative (N/-1). Table 2.1 shows the cost matrix for a traditional binary classifier which does not incorporate by default a reject option.

Actual\Predicted	$\widehat{P}$	$\widehat{N}$
P	C_{TP}	C_{FN}
N	C_{FP}	C_{TN}

Table 2.1 Cost matrix

When a reject option is included in the classifier, the new cost matrix as shown in table 2.2 will now have one additional column added which includes the cost of rejecting a positive sample and negative sample.

Actual\Predicted	$\widehat{P}$	$\widehat{N}$	$\widehat{R}$
P	C_{TP}	C_{FN}	C_{RP}
N	C_{FP}	C_{TN}	C_{RN}

Table 2.2 Cost matrix with reject option

In a cost-sensitive binary classification the Receiver Operating Characteristics (ROC) space has been an important tool in deciding the best operating point of the classifier. But the ROC space has been very poorly exploited to build a classifier with a reject option because the ROC space itself by default does not assume the classifier to reject any sample. Figure 2.1 shows the ROC space with the axis representation.

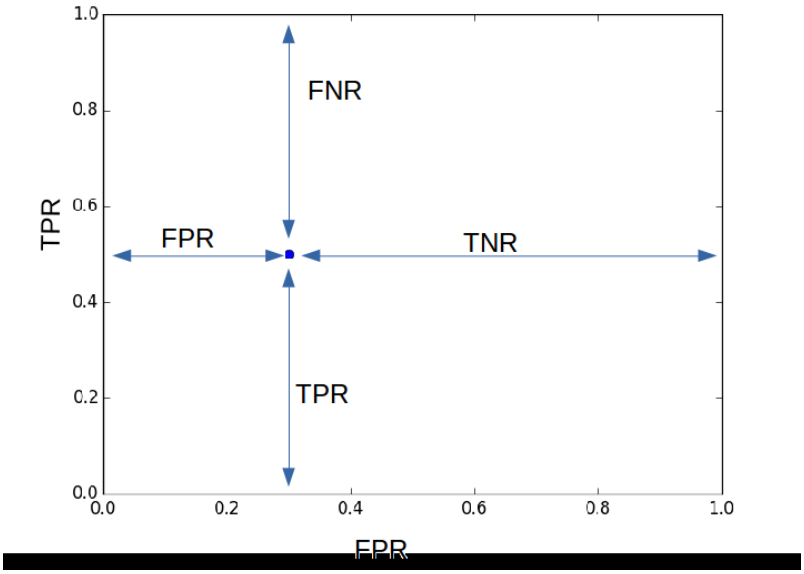

Figure 2.1 ROC space

The Area Under Curve (AUC) of the ROC is used as a measure to optimize the hyper-parameters in popular classifier algorithms such as Support Vector Machines (SVM). But this is not an optimal approach as the scaler criterion AUC considers the entire curve rather than only considering the operating point. Figure 2.2 clearly shows that classifier B performs much better than classifier A at the optimized operating point given by the slope of the iso-performance line.

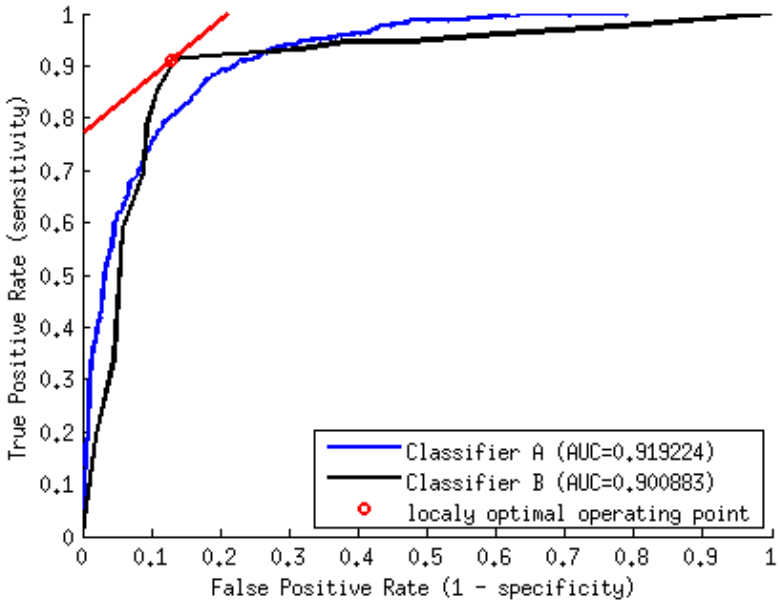

Figure 2.2 Classifier operating point

The ROC-Front method is an ensemble method specifically for cost sensitive classification which rationalize in replacing the traditional decision parameter optimization with a model selection approach. It is basically an ensemble of diverse cost-sensitive classifiers in the ROC space and given the cost model it selects the classifier that minimizes the cost of classification. Figure 2.3 shows the cost-sensitive non-dominated classifiers selected for the ensemble based on Pareto domination context where the classifiers that are not considered to be good for any cost sensitive models are rejected.

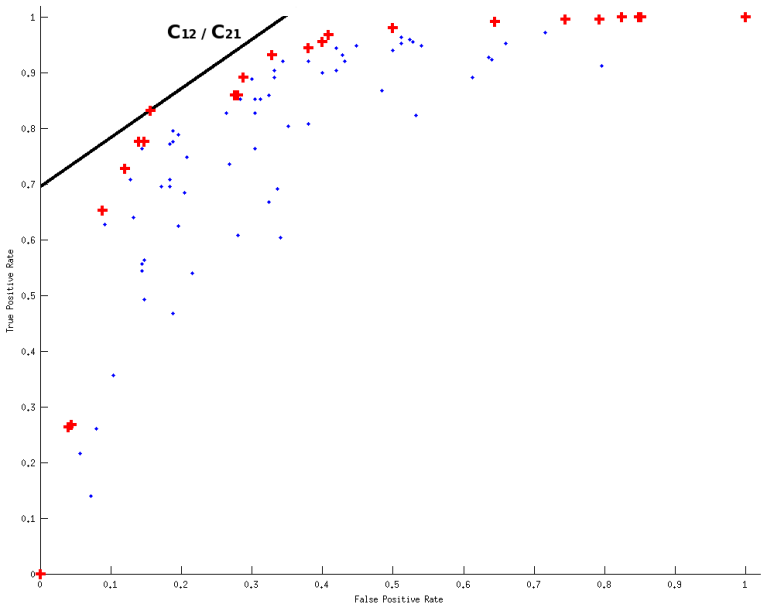

Figure 2.3 ROC of cost-sensitive classifiers. Blue dots are dominated classifiers and red dots are non-dominated classifiers

3. METHOD

The RFR method which is a ROC Front variant with a reject option and it works in a way to utilize the previously mentioned model based method to create an ensemble with cost-sensitive classifier pairs which selects the suitable pair based on the cost model.

Step 1: All the pairs of classifiers $H_{ij} = (h_i, h_j)$ are generated from the ROC Front and the prediction is based on the decision function given below.

$$\hat{y} = \begin{cases} P & if\ h_i(x) = P\ and\ h_j(x) = P \\ N & if\ h_i(x) = N\ and\ h_j(x) = N \\ R & else \end{cases}$$

Step 2: The classifier pairs are then projected on to the 4D ROC space where the dimensions corresponds to TPR, FPR, RPR and RNR, where RPR and RNR corresponds to Rejected Positive Rate and Rejected Negative Rate respectively. The point P(1,0,0,0) corresponds to the classifier pair which always predicts correctly. Any other point on the 4D ROC space is a cost-sensitive classifier pair with a reject option and the closer to point P the more accurate is the classification. The goal is to get a diverse set of classifiers spread across the 4D ROC space.

Step 3: A new ROC Front is built from the pairs of classifiers which are non-dominated in the Pareto-domination sense. That is the classifier pairs that can never be considered to do perform well at any cost-sensitive scenario are removed.

Now when the cost matrix is considered the ROC Front variant with a reject option denoted as RFR choses the best set of classifier by minimizing the below cost function.

$$L(H_{ij}, D) = p(P)\left(\sum_{j=1..3} C_{1j}R_{1j}\right) + p(N)\left(\sum_{j=1..3} C_{2j}R_{2j}\right)$$

where D is the given dataset

$p(P)$ and $p(N)$ are the prior probabilities of positive and negative sample

C_{1j} is the first row of cost matrix from table 2.2

C_{2j} is the second row of cost matrix from table 2.2

R_{1j} and R_{2j} are the corresponding rates on D

4. IMPLEMENTATION

Listed below are self-implementations of the algorithm specified in the paper using Matlab.

load_costmatrices

Script to randomly generate cost matrices for cost model 1, 4 and 8 which can be edited to generate as many cost models as required.

load_dataset

Script to load the dataset on which the training and testing is done.

load_hyperparameters

Script to load the hyperparameters for the Support Vector Machines (SVM).

init_script

Initialization script to load all required dataset, parameters, cost matrices and other variable needed for the main function.

PartitionForCV

Function to divide the given dataset into the number of partitions as specified by the k-fold variable 'F'.

CalculateConfusionMatrix

Function to calculate the confusion matrix for a particular k-fold dataset. The confusion matrix is similar to the cost matrix where the cost values are replaced by the corresponding rates.

CalculateConfusionMatrixMean

Function to calculate the mean of all the confusion matrices of the k-fold cross validation set.

Train

Function to train the SVM with the given parameters and the training data.

Predict

Function to predict the output using the trained SVM model and the given test data.

Main

The main script file that connects all the scripts and functions.

BestCombiningClassifier

Function to find the best SVM classifier combinations for all the given cost models.

5. RESULT

The RFR algorithm implemented as part of this project gives the best pair of SVM classifiers for the different cost models specified. It models the real world problem where the classifier takes into account the cases where it has to reject the samples in order to avoid the huge cost of misclassification. The main advantage of this model is that it always gives a cost-sensitive solution which is not always possible by many other cost-sensitive models that implements the reject option.