COST-SENSITIVE LEARNING OF CLASSIFICATION TREES, WITH APPLICATION TO IMBALANCED DATASETS

Magzhan Gabidolla[1] and **Arman Zharmagambetov**[2] and
Miguel Á. Carreira-Perpiñán[1], [1]EECS, UC Merced, [2]Meta, FAIR

1 Introduction

Many important practical applications involve a binary classification problem with imbalanced classes or asymmetric costs. Examples are fraud or spam detection or churn prediction. We focus on **decision trees**, which are widely recognized as among the most interpretable models. In these types of problems, optimizing the raw accuracy does not work well because it can largely ignore the low-cost or infrequent class. It is desirable to have control on the number of false positives or true positives. We formally propose the concept of *cost-optimal curve (COC)*. This defines a set of optimal accuracy classifiers as a function of the false positive level. We give an equivalent, penalized formulation which has the form of a weighted 0/1 loss and (with our new algorithm) is more amenable to optimization, although still NP-hard in general. We propose the first algorithm that directly tries to optimize this problem for classification trees, with the guarantee that the weighted 0/1 loss decreases monotonically at each iteration.

Work partially supported by NSF award IIS–2007147.

2 The ROC curve and the cost-optimal curve (COC)

The ROC curve:

- Is obtained by **postprocessing** a classifier through a threshold $t \in [0,1]$, so that it predicts the positive class if $p(y=+1|\mathbf{x}) > t$.
- Over a training set with N points this defines a set of at most $N+1$ classifiers, each corresponding to an ROC point (FP,TP).
- Does **not** produce a classifier that, having the desired FP rate, is **optimal** within its model class.

The Cost-Optimal Curve (COC):

- Aims to optimize:

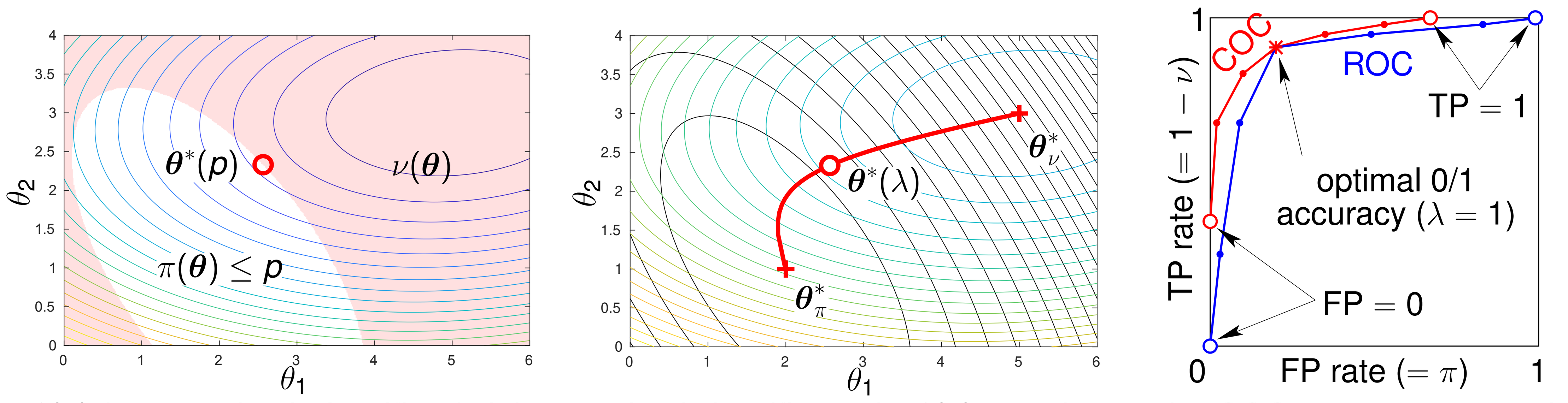
$$\min_{\boldsymbol{\theta}} \ \nu(\boldsymbol{\theta}) \quad \text{s.t.} \quad \pi(\boldsymbol{\theta}) \le p \qquad \text{with}$$

$$\nu(\boldsymbol{\theta}) = \sum_{n:\ y_n=+1}^{N} L(y_n, T(\mathbf{x}_n;\boldsymbol{\theta})), \quad \pi(\boldsymbol{\theta}) = \sum_{n:\ y_n=-1}^{N} L(y_n, T(\mathbf{x}_n;\boldsymbol{\theta})) \qquad (1)$$

where $L(\cdot,\cdot)$ is the 0/1 loss, ν is the FN rate, π is the FP rate.
- In practice, it solves the following unconstrained optimization:

$$\min_{\boldsymbol{\theta}} \ \nu(\boldsymbol{\theta}) + \lambda\,\pi(\boldsymbol{\theta}) \qquad (2)$$

where $\lambda \ge 0$. This objective function is a weighted 0/1 loss.
- Problems (1) and (2) have the same set of solutions, i.e., solving (1) for all $p \in [0,N]$ produces the same set of classifiers as solving (2) for all $\lambda \ge 0$ and hence the same COC curve. But solving (2) is easier than (1) in our case.
- **Dominates the ROC curve** (or any other curve using the same classifier family). That is, for any point (FP,TP) on the ROC curve there exists another point (FP',TP') on the COC curve with FP' $\le$ FP and TP' $\ge$ TP.



$\boldsymbol{\theta}^*(p)$ is an optimal classifier (minimizing ν) with an FP rate of at most p. The infeasible set is in pink. The contours of ν and π are in color and black, respectively.

optimal classifier path $\boldsymbol{\theta}^*(\lambda)$ over the cost λ, i.e., minimizing $\nu + \lambda\pi$, from $\boldsymbol{\theta}^*_\nu = \boldsymbol{\theta}^*(0)$ to $\boldsymbol{\theta}^*_\pi = \boldsymbol{\theta}^*(\infty)$. The contours of ν and π are in color and black, respectively.

the COC curve corresponding to the optimal classifier path and the ROC curve (assuming as base classifier that for $\lambda = 1$).

Figure: Illustration of the COC curve for a classifier with parameters $\boldsymbol{\theta} \in \mathbb{R}^2$, FP rate $\pi(\cdot)$, FN rate $\nu(\cdot)$.

input: training set $\{\mathbf{x}_n, \mathbf{y}_n \in \{-1,1\}\}_{n=1}^N$, depth of the tree Δ, regularization parameter $\alpha \ge 0$, schedule parameter $\beta > 1$. Set the base cost $\lambda_0 = \frac{N_+}{N_-}$.
$T_0(\cdot;\boldsymbol{\Theta})$ = TAO on a random tree of depth Δ with cost $\lambda = \lambda_0$
$T_0^-(\cdot;\boldsymbol{\Theta}) = T_0(\cdot;\boldsymbol{\Theta})$, $\lambda = \lambda_0$, $i = 0$
repeat
  $\lambda \leftarrow \beta\lambda$, $i \leftarrow i+1$
  $T_i^-(\cdot;\boldsymbol{\Theta})$ = TAO on $T_{i-1}^-(\cdot;\boldsymbol{\Theta})$ as initial tree with cost λ
until false positives by $T_i^-(\cdot;\boldsymbol{\Theta})$ is zero
$T_0^+(\cdot;\boldsymbol{\Theta}) = T_0(\cdot;\boldsymbol{\Theta})$, $\lambda = \lambda_0$, $i = 0$
repeat
  $\lambda \leftarrow \lambda/\beta$, $i \leftarrow i+1$
  $T_i^+(\cdot;\boldsymbol{\Theta})$ = TAO on $T_{i-1}^+(\cdot;\boldsymbol{\Theta})$ as initial tree with cost λ
until false negatives by $T_i^+(\cdot;\boldsymbol{\Theta})$ is zero
return all trained trees $\{T_i^-(\cdot;\boldsymbol{\Theta})\}_i \cup \{T_i^+(\cdot;\boldsymbol{\Theta})\}_i$

Figure: Pseudocode of COC with decision trees

3 Tree Alternating Optimization (TAO)

To realize the advantages of the COC curve one needs to optimize (2), which uses a weighted 0/1. Recently, an algorithm has been proposed (*Tree Alternating Optimization (TAO)*), which does optimize a global loss over a parametric tree (axis-aligned or oblique). We extend TAO to handle a weighted 0/1 loss objective:

$$E(\boldsymbol{\Theta}) = \sum_{n:\ y_n=+1}^{N} L(y_n, T(\mathbf{x}_n;\boldsymbol{\Theta})) + \lambda \sum_{n:\ y_n=-1}^{N} L(y_n, T(\mathbf{x}_n;\boldsymbol{\Theta})) + \alpha \sum_{i\in\mathcal{D}} \phi_i(\boldsymbol{\theta}_i) \qquad (3)$$

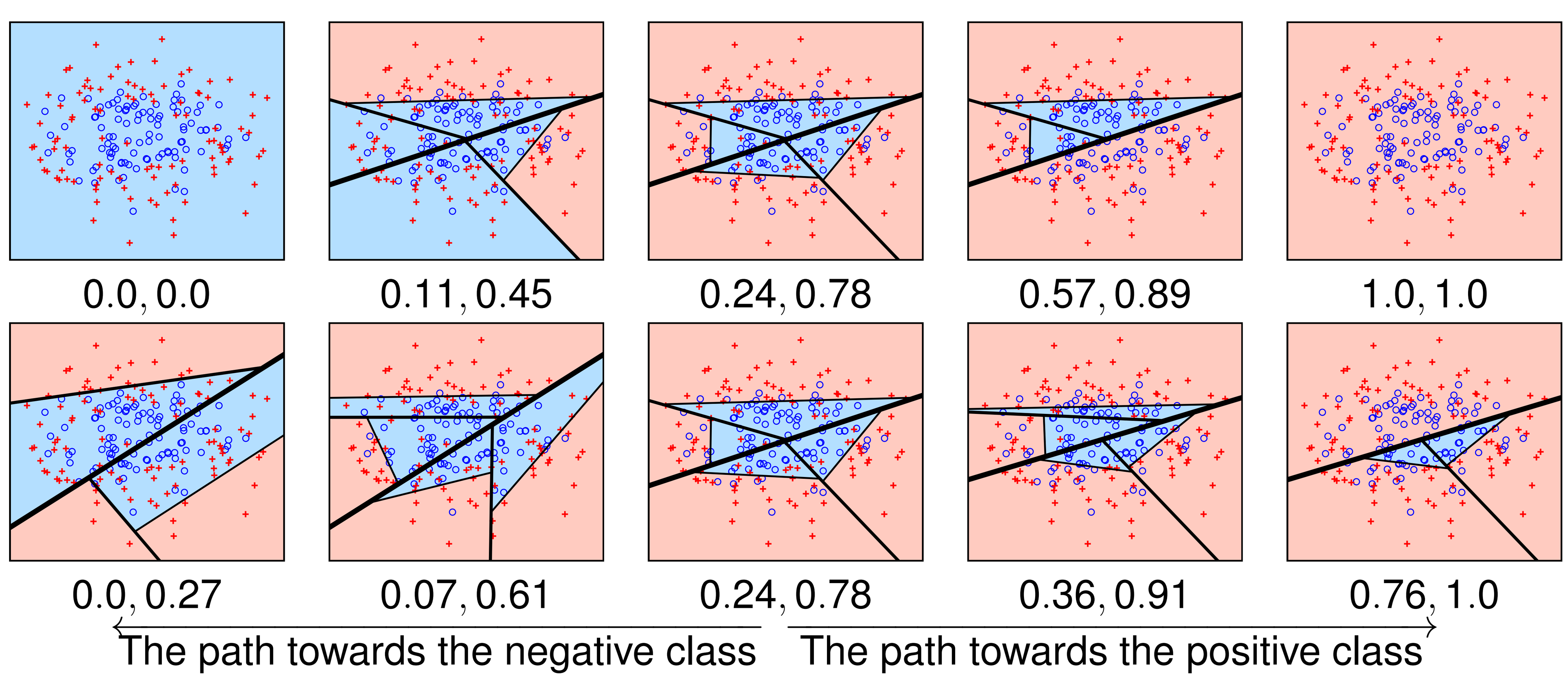
The algorithm is based on 3 theorems:

- **separability condition**: objective function (3) separates over any set of non-descendant nodes (e.g. all nodes at the same depth), those can be optimized independently and in parallel.
- **Optimizing a decision node reduces** to a simpler problem of **a weighted 0/1 loss binary classification** over the node weights. In practice, we solve it using convex surrogate, logistic regression.
- **Optimizing a constant label leaf** is simply solved by setting the label to the weighed majority class.

input initial $T(\cdot,\boldsymbol{\Theta})$ of depth Δ, training set $\{\mathbf{x}_n, \mathbf{y}_n \in \{-1,1\}\}_{n=1}^N$, cost of false positives λ, regularization parameter $\alpha \ge 0$
repeat
  for $d = \Delta$ **to** 0 **do**
    for all nodes *node* at depth d
      if *node* is a leaf
        set the label of the *node* to the most costly class in the reduced set
      else
        fit a weighted 0/1 binary classifier where weights come from the costs
until convergence
return tree $T(\cdot;\boldsymbol{\Theta})$

Figure: TAO pseudocode for cost-sensitive learning

3 Toy 2D illustration



4 Imbalanced classification