In brief...

- Boosting is a successful family of ensemble learning algorithms. However, boosting algorithms despite being good classifiers – or rather because of it – exhibit poor probability estimation, which also makes them poor at cost-sensitive / imbalanced class learning.
- In recent work we established that all cost-sensitive boosting variants in the literature (1997-2016) fail to satisfy all desirable theoretical properties. The few that do satisfy all such properties, do so only after calibration of their scores.
- We proposed reserving part of the dataset to calibrate the scores of the ensemble and shifting the decision threshold accordingly to account for the cost / class imbalance – fast, easy to implement, satisfies all properties, can easily adjust to changes in costs / imbalance, matches or outperforms any other method on a wide range of cost-sensitive tasks.
- Current work focuses on extending these ideas to the online learning case. The main complication is that we can no longer split our data into a training & a calibration set; we need to decide for each minibatch whether to use it for training the ensemble or the calibrator.

Motivation

We want classifier scores to be good probability estimates:

- To quantify our uncertainty over their predictions & know how much to trust them.
- To make better cost-sensitive decisions (see study case below).

We want online learning algorithms:

- To process streaming data.
- To deal with situations where the problem (e.g. data distribution) changes over time, possibly in an adversarial fashion.
- To efficiently process big amounts of data, even when they all are available at once.

Why boosting:

- Very successful in classification / regression tasks – 60% of the winning Kaggle entries were based on boosting.
- Very good classifier but very poor at estimating probabilities (no coincidence).
- Very rich theory motivating it & connecting it to other areas of machine learning.

A study case: cost-sensitive boosting

All cost-sensitive boosting variants in the literature (1997-2016) fail to satisfy all desirable theoretical properties of if (i) having internally consistent steps (ii) making optimal decisions under decision theory, (iii) preserving the relative importance of the classes during training. (iv) having calibrated probability estimates.

Method | FGD-consistent | Cost-consistent | Asymmetry-preserving | Calibrated estimates
--- | --- | --- | --- | ---
AdaBoost (Friedol & Schapire 1997) | | | | 
AdaCSt (Fan et al. 1998) | | | | 
AdaCost (Jh 2006) | | | | 
CSB0 (2006) | | | | 
CSB1 (2006) | | | | 
CSB2 (2006) | | | | 
AdaC1 (Yu et al. 2005, 2007) | | | | 
AdaC2 (Yu et al. 2005, 2007) | | | | 
AdaC3 (Yu et al. 2005, 2007) | | | | 
CSACh (Mauadd & Wahid 2007, 2011) | | | | 
AdaDB (Yuand&Vapnik & Aase & Carpenter, 2011) | | | | 
AdaMEC (Yu et al. 2005) | | | | 
AdaMEC (2006) | | | | 
CCG-M (Flach & Konstantinov 2010, 2013) | | | | 
Cry-Ad (Yuo & Jones 2006) | | | | 

Boosting probability estimates are inherently uncalibrated. It minimizes monotonically decreasing loss functions of the margin. Maximizing the margin increases the confidence of the classifier in its predictions, which has been connected to good generalization in classification. But, the generated probability estimates tend to 0 or 1 as the margins increase. What makes boosting a good classifier, also makes it a bad probability estimator!

Once calibrated, AdaMEC, CGAda & AsymAda (different approximations of same model) satisfy all properties. Based on theoretical soundness, flexibility, simplicity & results our suggestion for practitioners is Calibrated AdaMEC:

Results

Learning curves of avg-log-loss per 10% of minibatches seen, for ensembles of M=10 Naïve Bayes weak learners – bandit policy used: UCB1-Impoved (Auer et al., 2002)

Similar results for other bandit policies, other weak learners, regularized weak learners, varying ensemble sizes, presence of inherent non-stationarity

Some calibration (even worse naive) is better than no calibration

Bandit policies are online, fast, at least as good as ‘best naive’ + adaptive to non-stationarity

Easy to adapt to other problems (e.g. cost-sensitive learning)

Robust to ensemble/calibrator hyperparameters

Extensions: e.g. adversarial, contextual, more actions, refine calibration, ...

Bandit problem: set of actions (arms) – on each round we choose one; each action associated with reward distribution; each time an action is taken we sample its reward distribution.

Goal: find the sequence of actions that minimize cumulative regret (exploration vs. exploitation)

Bandit Policies examined:

- Thompson sampling: Bayesian way of updating reward distribution; Assume rewards Gaussian, start with Gaussian prior, then update using self-conjugacy of Gaussian distribution
- Take action with highest posterior reward
- UCB policies: ‘Optimism in the face of uncertainty’ Choose not the action with best expected reward, but that with highest upper bound on reward

Acknowledgements: NN & GB were supported by the EPSRC grants [EP/028090/1] & [EP/L000725/1]; NN acknowledges the support of the EPSRC Doctoral Prize Fellowship