Predicting accurate probabilities with a ranking loss

Aditya Krishna Menon, Xiaoqian Jiang, Shankar Vembu, Charles Elkan, Lucila Ohno-Machado

{akmenon, x1jiang, elkan, lucila}@ucsd.edu; shankar@utoronto.ca

From classification to probability estimation

1

Classically, in supervised learning we aim to predict the label of a future test example. In many practical applications, though, we need to predict probabilities of labels, e.g.:

(a) Taking actions to maximize expected utility, which is naturally handled by estimating the probability of reward given an action;

(b) Feeding in predictions to a meta-classifier, be it another automated system or a human expert like a physician, where it is essential to estimate the confidence in predictions;

(c) Non-standard learning tasks, like positive and unlabelled examples, are solvable (with assumptions) if we estimate probabilities

Theory: proper losses

2

The generalization error of a model $\hat{y} : \mathcal{X} \to \mathbb{R}$ wrt a loss $\ell : [0,1] \times \{0,1\} \to \mathbb{R}_+$ is

$$\mathcal{E}(\hat{y}(\cdot)) = \mathbb{E}_{x,y}\ell(\hat{y}(x), y) = \mathbb{E}_x L_\ell(\hat{y}(x), \eta(x)) \quad \text{where} \quad \eta(x) = \Pr[y = 1|x]$$

If the minimizer of $L_\ell(\hat{y}, \eta)$ for a fixed η is $\hat{y} = \eta$, then the model with best generalization error is $\hat{y}(x) = \eta(x)$. We call a loss satisfying this proper.

Existing approach #1: proper loss minimization

3

The simplest scheme to obtain probability estimates is optimizing a proper loss between the model predictions and the true labels. Generally, a regularizer Ω is added to prevent overfitting:

$$\frac{1}{n}\sum_{i=1}^n \ell(\hat{y}(x_i; w), y_i) + \lambda\Omega(w)$$

This covers linear and logistic regression, neural networks, *et cetera*. Possible failure modes include:

(a) Misspecification of the link or scoring function

(b) Finite sample effects, such as bias of learned *w*

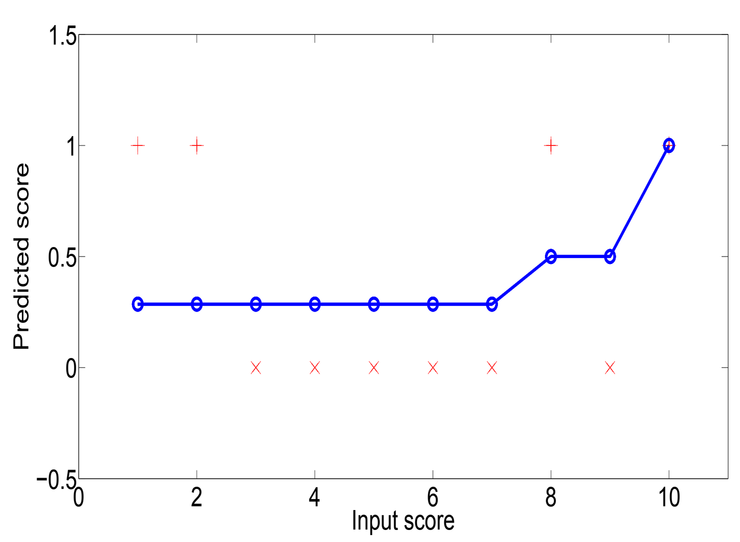
(c) Including a regularization term makes the loss function non-proper

Existing approach #2: post-processing

4

Another option is to take the scores of some input model - not necessarily one optimizing a proper loss - and then post-process these in some way. Isotonic regression (IR) is one such scheme. Here, we assign a score to each example that preserves the ordering of input model scores, and minimizes squared loss (which is proper) to the observed labels:

$$\min_{\tilde{y}_1, \ldots, \tilde{y}_n} \sum_{i=1}^n (\tilde{y}_i - y_i)^2 : \hat{y}_j < \hat{y}_k \implies \tilde{y}_j \leq \tilde{y}_k$$



A question arises: what model should we use as input to isotonic regression?

Our approach: ranking loss + isotonic regression

5

The result of isotonic regression is the same for any two input sets of scores that rank examples in the same order. This suggests that to maximize the accuracy of estimates from isotonic regression, we should use the input model with maximal pairwise ranking performance.

In particular, we look to maximize the area under the ROC curve (AUC), being the probability of a randomly drawn positive having a higher score than a randomly drawn negative:

$$\text{AUC} = \Pr_{(x_1,y_1),(x_2,y_2)}[\hat{y}(x_1) \geq \hat{y}(x_2) | y_1 = 1, y_2 = 0]$$

To maximize this, consider the regularized empirical convex approximation

$$F(w) = \frac{\lambda}{2}||w||^2 + \frac{1}{n_+ n_-}\sum_{i,j=1}^n y_i(1 - y_j)\ell(w^T(x_i - x_j), 1)$$

This pairwise ranking objective is the basis of e.g. SVMRank. It may be optimized efficiently using stochastic gradient descent: we need to pick a random (positive, negative) pair. The resulting model will have good ranking performance in an AUC sense. We then apply isotonic regression to these scores to get probability estimates. In summary, our approach is:

(1) for $t = 1 \ldots T$
- Pick random examples $(x_1, +)$ and $(x_2, -)$
- Update *w* based on gradient of *F*(*w*) above

(2) Apply isotonic regression on resulting scores $\{w^T x_i\}_{i=1}^n$

Our approach has good ranking and regression performance simultaneously. An existing model for this regime is the Combined Regression and Ranking (CRR) framework [Sculley, KDD'10] which optimizes, for a fixed scalar α

$$\alpha\text{Square-Loss} + (1 - \alpha)\text{Rank-Loss}$$

Compared to our approach, in CRR the regression component is parametric, and hence limited in what it can model. Further, the overall objective is not proper (even without regularization), nor does it exactly maximize the AUC, because of the linear tradeoff. Our approach manages to overcome these two concerns by dealing with them sequentially.

Justification of approach

6

We establish a consistency result for the model: if $\Pr[y = 1|x] = f(w^T x)$ for some monotone *f*(.), then we will recover the probability distribution in the limit of infinite samples. This class of probability distributions is known as the single-index model family.

The proof of this claim relies on the following ingredients:

(a) The AUC is maximized by predicting a monotone transform *c*(.) of $\Pr[y = 1|x]$; see [Clemencon *et al*, AOS '06].

(b) Minimizing *F*(*w*) as above will recover the true probability [Clemencon *et al*, AOS '06].

(c) Isotonic regression on top of an optimal ranking recovers the true probability.

To show (c), we note that isotonic regression outputs calibrated probability estimates, so that

$$\Pr[y = 1|\hat{y} = s] = s$$

But using (a), and the fact that the predictions $\Pr[y = 1|x]$ are themselves calibrated,

$$\Pr[y = 1|\eta = c^{-1}(s)] = c^{-1}(s) = s$$

Therefore, it must be the case that we have recovered the true probabilities.

Experimental setup

7

We run experiments on one synthetic and the following three real-world datasets:

(a) KDDCup '98, where the goal is to maximize utility by contacting customers estimated to be receptive to a solicitation;

(b) GCAT, a document collection with positive and unlabelled examples only;

(c) Hospital Discharge, a medical informatics dataset where confidence scores are essential for use in a meta-classifier

The methods we compare are linear (LinReg) and logistic (LogReg) regression, with and without post-processing by isotonic regression (+ IR), the combined regression and ranking model (CRR), and our approach (Rank + IR).

Experimental results

8

We generate synthetic data where the link function is a capped step function:

$$\Pr[y = 1|x] = a + (1 - a)\mathbf{1}[w_0^T x > 0]$$

for some fixed w_0. Note that as $a \to 0$, this approaches a standard step function, and the sigmoid used in logistic regression is a reasonable fit. As $a \to 1/2$, the step function occupies a very thin band, and the sigmoid is a bad fit.

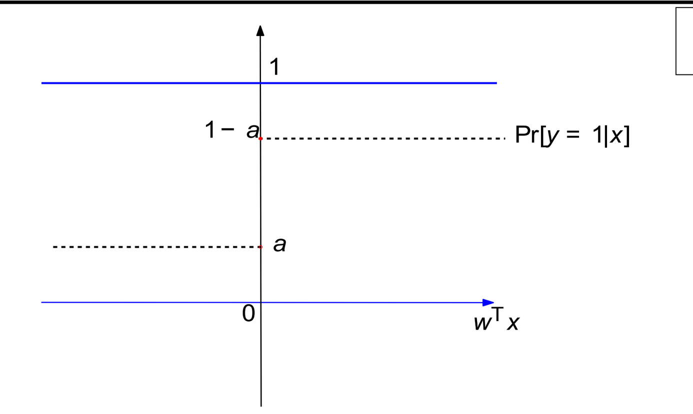


Fig 1: Illustration of link function for synthetic data.

We see that isotonic regression significantly improves the probability estimates of logistic regression. (Thus, even mature methods like logistic regression may be defeated by misspecification.) Our method further improves the quality of these estimates for a range of *a* values.

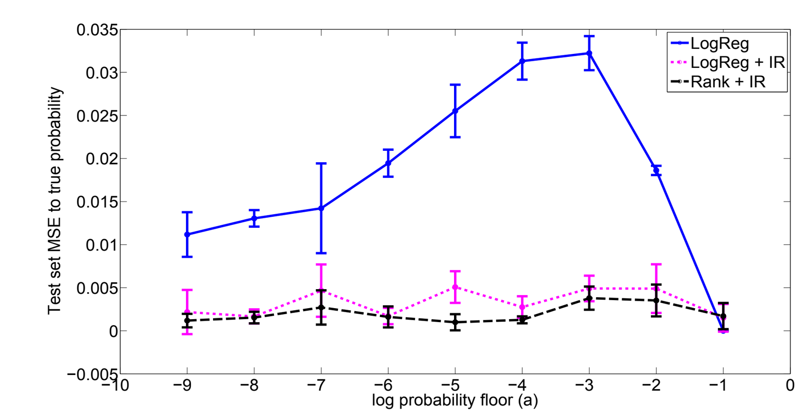


Fig 2: Results on synthetic dataset.

On the real-world datasets, we generally achieve best of both worlds performance: good ranking according to AUC, and good regression according to MSE or utility. Observe for example the $300 improvement in utility over logistic regression on the KDDCup '98 dataset. Generally, logistic regression processed with isotonic regression also performs well. The good ranking performance of logistic regression has been an area of recent theoretical study [Kotlowski *et al*, ICML '11].

Table 1: Test set results on KDDCup '98 dataset.

Method	Test set profit	AUC
LinReg	\$12,479.12	0.6157
LinReg + IR	\$13,142.72	0.6157
LogReg	\$13,338.22	0.6160
LogReg + IR	\$12,861.88	0.6160
CRR	\$13,249.60	0.6162
Rank + IR	\$13,671.44	0.6162

Table 2: Average test split results on GCAT dataset.

Method	MSE	AUC
LinReg	0.0550 ± 0.0015	0.9824 ± 0.0017
LinReg + IR	0.0478 ± 0.0021	0.9823 ± 0.0014
LogReg	0.0579 ± 0.0021	0.9836 ± 0.0007
LogReg + IR	0.0423 ± 0.0024	0.9836 ± 0.0007
CRR	0.0557 ± 0.0020	0.9825 ± 0.0015
Rank + IR	0.0419 ± 0.0021	0.9831 ± 0.0005

Table 3: Average test split results on Hospital Discharge dataset.

Method	MSE	AUC
LinReg	0.0461 ± 0.0000	0.6987 ± 0.0013
LinReg + IR	0.0465 ± 0.0002	0.6987 ± 0.0013
LogReg	0.0458 ± 0.0001	0.7066 ± 0.0009
LogReg + IR	0.0461 ± 0.0001	0.7066 ± 0.0009
CRR	0.0461 ± 0.0000	0.7045 ± 0.0016
Rank + IR	0.0460 ± 0.0003	0.7081 ± 0.0021