Classification with Rejection Based on Cost-sensitive Classification

Nontawat Charoenphakdee[1,2], Zhenghang Cui[1,2], Yivan Zhang[1,2], Masashi Sugiyama[2,1]

The University of Tokyo[1], RIKEN AIP[2]

ICML2021

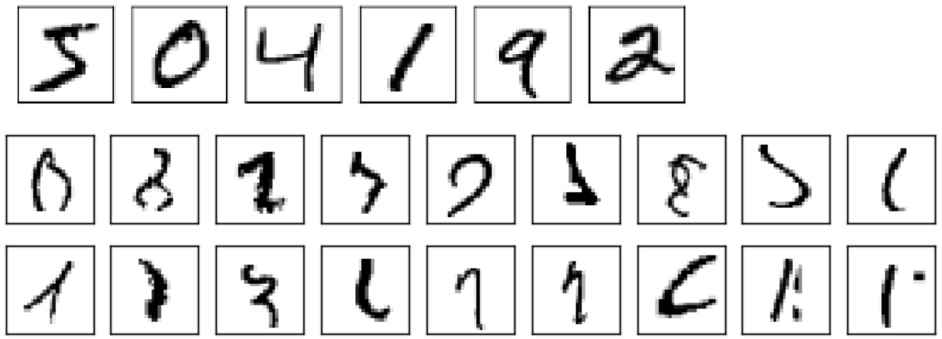
東京大学
THE UNIVERSITY OF TOKYO

RIKEN

Mistake in predictions can be (very) harmful

Medical diagnosis

Guess the number (0-9):



MNIST dataset (Lecun, 1998)

Always answering is **prone to misclassification**.

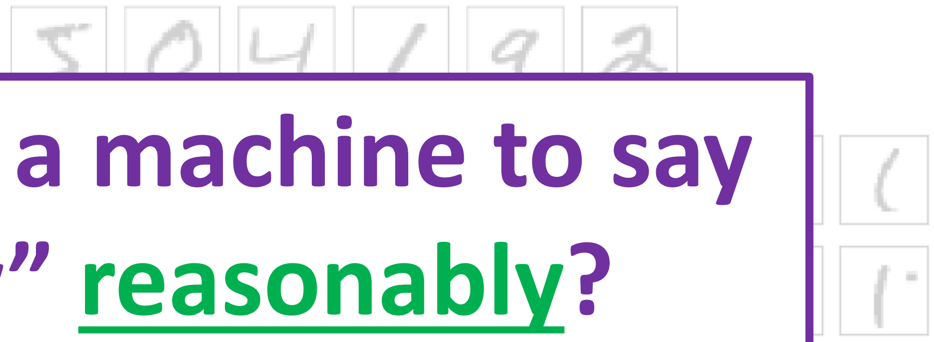
Saying **"I don't know"** can **reduce misclassification**.

https://www.iconfinder.com/icons/5692300/ai_doctor_medical_nurse_robot_stethoscope_icon

Mistake in predictions can be (very) harmful

Medical di

Guess the number (0-9):



MNIST dataset (Lecun, 1998)

How to teach a machine to say "I don't know" reasonably?

Always answering is **prone to misclassification**.

Saying **"I don't know"** can **reduce misclassification**.

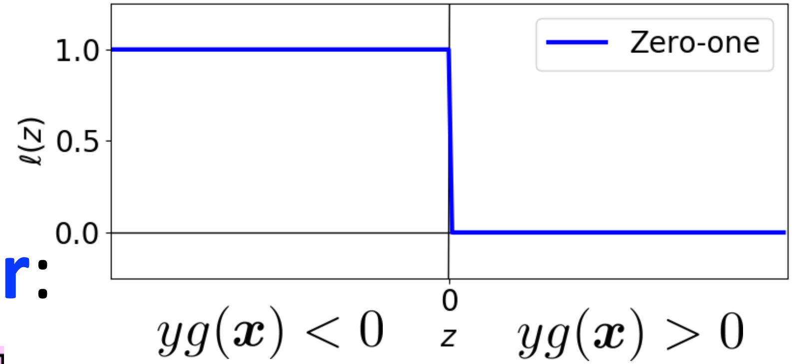
https://www.iconfinder.com/icons/5692300/ai_doctor_medical_nurse_robot_stethoscope_icon

Warmup: binary classification

$y \in \{-1, 1\}$: Label
$g\colon \mathbb{R}^d \to \mathbb{R}$: Prediction function
$\boldsymbol{x} \in \mathbb{R}^d$: Feature vector
$\ell\colon \mathbb{R} \to \mathbb{R}$: Margin loss function
$z = yg(\boldsymbol{x})$: Margin

- **Given**: Training input-output pairs:

$$\{\boldsymbol{x}_i, y_i\}_{i=1}^n \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x}, y)$$



$yg(\boldsymbol{x}) < 0$ **different sign**

$yg(\boldsymbol{x}) > 0$ **same sign**

- **Goal**: Find g that minimizes the **expected error**:

$$R^{\ell_{0\text{-}1}}(g) = \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim p(\boldsymbol{x},y)} [\ell_{0\text{-}1}(yg(\boldsymbol{x}))]$$

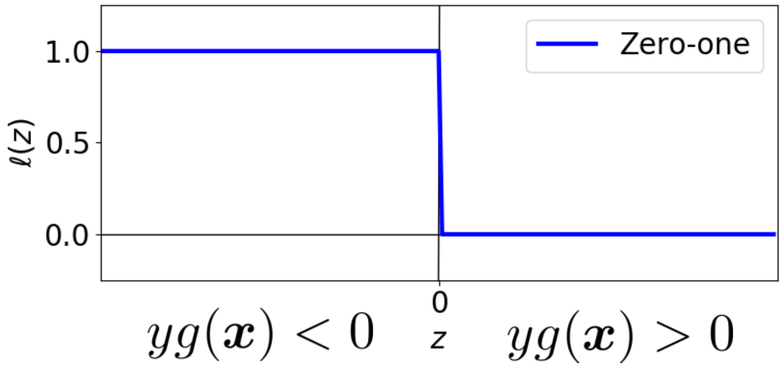
No access to distribution: cannot minimize the expected error directly.

- Instead, we minimize the **empirical error** (Vapnik, 1998):

$$\hat{R}^{\ell_{0\text{-}1}}(g) = \frac{1}{n}\sum_{i=1}^n \ell_{0\text{-}1}(y_i g(\boldsymbol{x}_i))$$

Zero-one loss and its surrogates

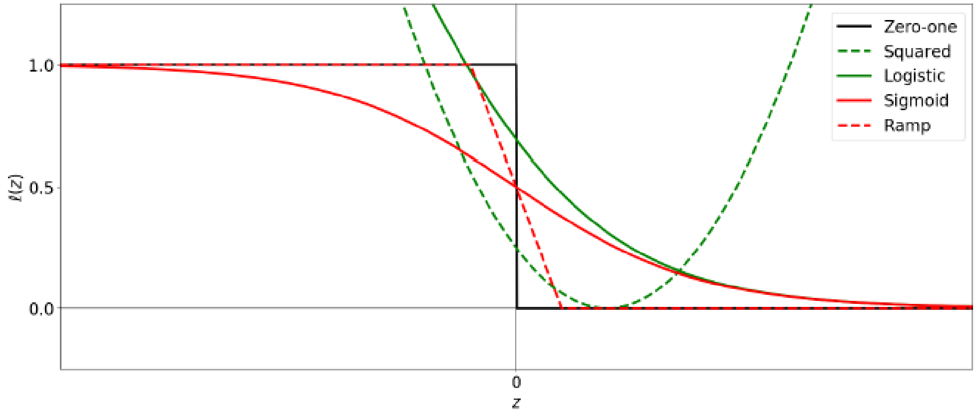
Zero-one loss



$yg(\boldsymbol{x}) < 0$ $yg(\boldsymbol{x}) > 0$

different sign **same sign**

Surrogate losses



$$\hat{R}^{\ell}(g) = \frac{1}{n}\sum_{i=1}^{n} \ell\left(y_i g(\boldsymbol{x}_i)\right)$$

Minimizing $\hat{R}^{\ell_{0\text{-}1}}$ is NP-hard even for simple model.
(Ben-david+, 2003; Feldman+, 2012)

Surrogate losses that are easier to minimize are used in practice.

Classification-calibration ensures that minimizing R^{ℓ} yields good g for $R^{\ell_{0\text{-}1}}$
(Zhang, 2004; Bartlett+, 2006)

But zero-one loss does not concern rejection…

From *zero-one loss* to *zero-one-c loss*

(Chow 1957, 1970)

Define a rejection cost $c \in (0, 0.5]$

$g : \mathbb{R}^d \to \mathbb{R}$	: Prediction function
$r : \mathbb{R}^d \to \{0, 1\}$	: Rejection function

Zero-one-c loss

$$\ell_{0\text{-}1\text{-}c}(y, r(\boldsymbol{x}), g(\boldsymbol{x})) = \begin{cases} c & \text{if } r(\boldsymbol{x}) = 0 \\ \underbrace{\ell_{0\text{-}1}(yg(\boldsymbol{x}))}_{\text{zero-one loss}} & \text{otherwise} \end{cases}$$

Rejection comes with **rejection penalty** (less than **misclassification penalty**).

A classifier has an incentive to prefer **rejection** over **misclassification**

How to solve this problem?

Confidence-based approach

(Chow+ 1957, 1970; Yuan+, JMLR2010; Ni+, NeurIPS2019)

Knowing $p(y|x)$ is sufficient

$$g^*(\boldsymbol{x}) = p(y=1|\boldsymbol{x}) - \frac{1}{2}$$
$$r^*(\boldsymbol{x}) = \mathbb{1}_{[\max_y p(y|\boldsymbol{x}) - (1-c)]}$$

$g\colon \mathbb{R}^d \to \mathbb{R}$: Prediction function
$r\colon \mathbb{R}^d \to \{0,1\}$: Rejection function

(Chow 1957, 1970)

Pros: Straightforward to use in the multi-class case.

Cons: However, in general, surrogate losses must be able to estimate $p(y|\boldsymbol{x})$

Strictly stronger requirement than **classification-calibration**!

(Reid+ JMLR2010)

With deep learning, **accuracy is dramatically improved** but the **prediction confidence is no longer accurate**.

(Guo+, ICML2017; Thulasidasan, NeurIPS2019; Hein+, CVPR2019; Vasudevan+, ICASSP2019; Jagannatha+, ACL2020)

Classifier-rejector approach

(Cortes+, ALT2016, NeurIPS2016)

Train r and g simultaneously.

$\mathcal{H}$: Prediction function class
$\mathcal{R}$: Rejection function class

Goal: find $(r, g) \in \mathcal{H} \times \mathcal{R}$ that minimizes

$$\hat{R}^{\ell_{\text{0-1-}c}}(r, g) = \frac{1}{n} \sum_{i=1}^{n} \ell_{\text{0-1-}c}(y_i, r(\boldsymbol{x}_i), g(\boldsymbol{x}_i))$$

Limited loss choice (only exponential and max-hinge) for **binary case.**

(Cortes+ ALT2016, NeurIPS2016)

The multiclass extension of Cortes+ does not work theoretically and experimentally performed worse than confidence-based approach

(Ni+, NeurIPS2019)

Proposal: Cost-sensitive approach

Binary cost-sensitive classification (Scott, 2012)

Binary classification where

false positive penalty $\neq$ false negative penalty

Let $\alpha \in (0,1)$ be false positive cost and $1-\alpha$ be false negative cost

Ordinary classification: $\alpha = 0.5$

The solution of this problem is

$$\mathrm{sign}[p(y=+1|\boldsymbol{x})-\alpha]$$

Loss requirement: ***classification-calibration***

Solving one cost-sensitive classification means knowing if $p(y=+1|\boldsymbol{x}) > \alpha$

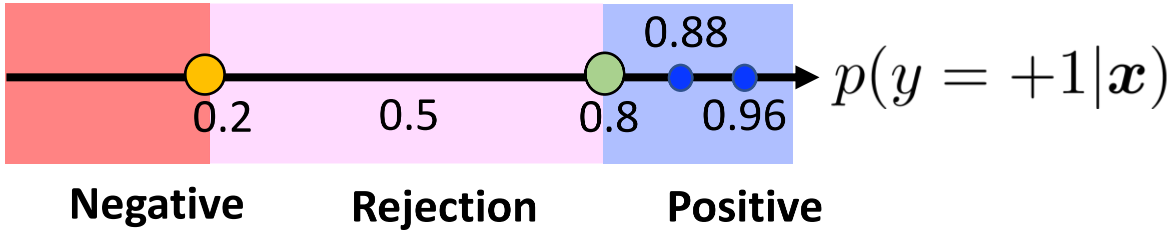
Cost-sensitive approach: motivation

Consider optimal decision rule for the binary case (Chow, 1970)

$$h^*(\boldsymbol{x}) = \begin{cases} \text{Positive} & p(y=+1|\boldsymbol{x}) > 1-c, \\ \text{Reject} & c \le p(y=+1|\boldsymbol{x}) \le 1-c, \\ \text{Negative} & p(y=+1|\boldsymbol{x}) < c, \end{cases}$$

We only need to know:

1. $p(y=+1|\boldsymbol{x}) > 1-c$
2. $p(y=+1|\boldsymbol{x}) < c$



Example: if c = 0.2, if we know $p(y=+1|\boldsymbol{x}) > 0.8$, it is ***unneeded to know its exact value***.

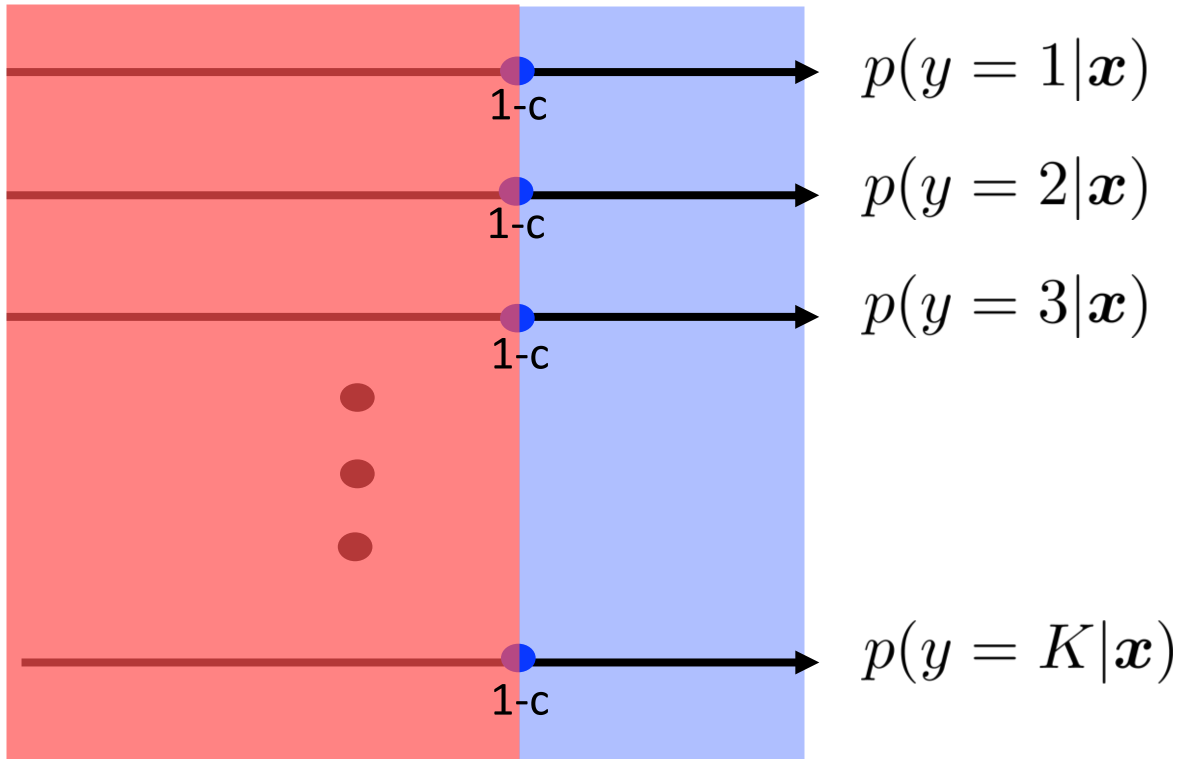
Solving cost-sensitive classification can validate if $p(y=1|\boldsymbol{x}) > \alpha$

Learn two cost-sensitive classifiers for $\alpha = c$ **and** $\alpha = 1-c$

Connecting cost-sensitive classification to classification with rejection.

Extension to multiclass scenario is simple

$$\mathcal{L}_{\mathrm{CS}}^{c,\phi}(\boldsymbol{g};\boldsymbol{x},y) = c\phi\big(g_y(\boldsymbol{x})\big) + (1-c)\sum_{y'\neq y}\phi\big(-g_{y'}(\boldsymbol{x})\big).$$



Predict if:

1. **Only one** classifier returns positive

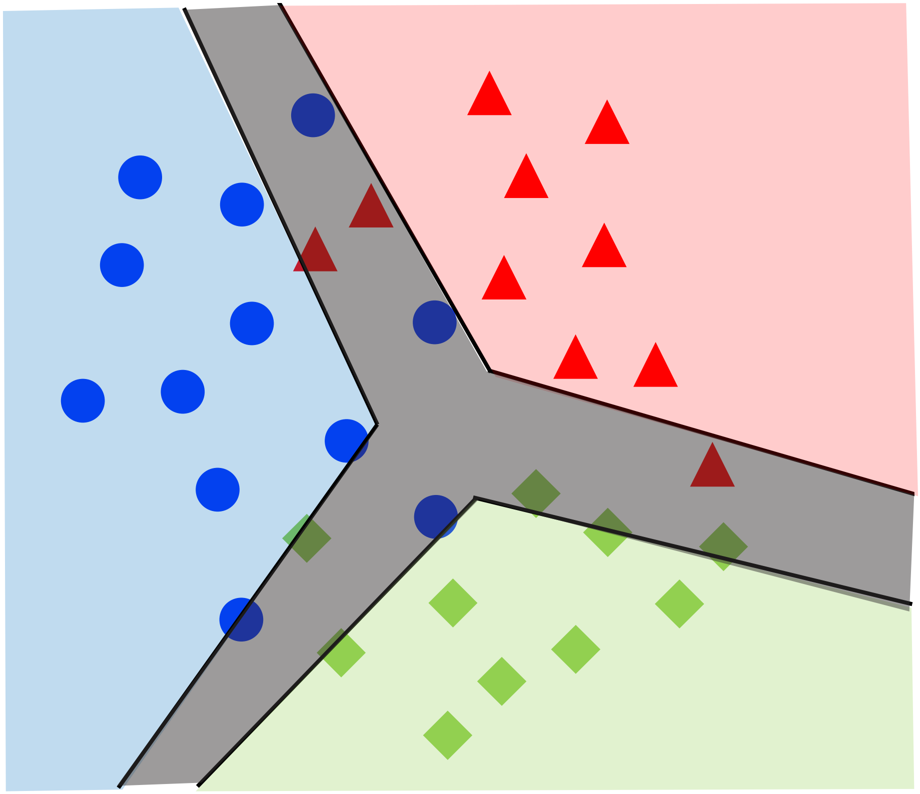
Reject if:

1. All classifiers return negative
2. **More than one** classifier return positive

Learn **K one-vs-rest cost-sensitive binary classifiers** with $\alpha = 1 - c$

Can be learned at once by learning a K-dimensional output function

Interpretation: cost-sensitive approach

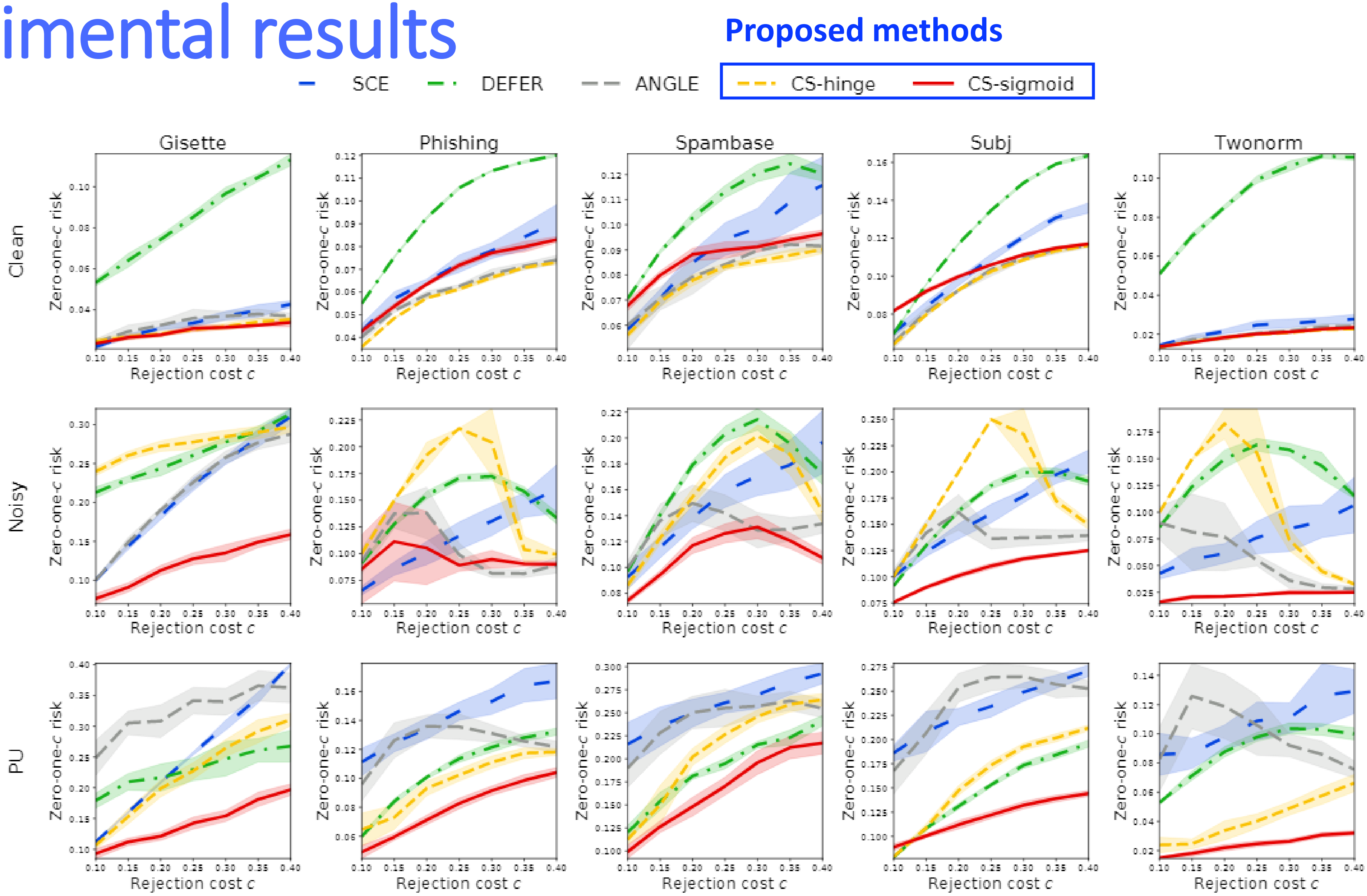


1. Learn K binary cost-sensitive classifiers
2. Reject if:
   - All classifiers predict negative
   - More than one classifier predicts positive

Loss requirement: ***classification-calibration***

A novel approach with flexible loss choices!

Experimental results



CS-hinge works well in classification from clean labels (Clean)

CS-sigmoid works well in classification from noisy labels (Noisy) and classification from positive and unlabeled data (PU)

Conclusions

Cost-sensitive approach: an approach for classification with rejection based on cost-sensitive classification, which

1. can avoid estimating class-posterior probabilities
2. allows a flexible choice of losses including non-convex ones
3. is applicable to both binary and multiclass cases
4. is theoretically justifiable for any classification-calibrated loss.