One-sided Support Vector Regression for Multiclass Cost-sensitive Classification

Han-Hsing Tu and Hsuan-Tien Lin

National Taiwan University

June 23, 2010

Binary Cost-sensitive Classification

?

cold-infected

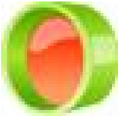
healthy

actual \ predicted	cold	healthy
cold	0	C_{-1}
healthy	C_1	0

binary, cost-matrix based

Multiclass Cost-sensitive Classification

?



H1N1-infected

cold-infected

healthy

error measure = society cost

actual \ predicted	H1N1	cold	healthy
H1N1	0	1000	**100000**
cold	100	0	3000
healthy	100	30	0

- human doctors consider costs of decision
- want computer-aided diagnosis to behave similarly

multiclass, cost-matrix based

From Cost Matrix to Cost Vector

with actual underlying status

prediction	H1N1	cold	healthy
society cost	0	1000	**100000**

- only a "row" needed per example: cost vector $\mathbf{c}$
  - an H1N1 patient: $\mathbf{c} = (0, 1000, 100000)$
  - a healthy patient: $\mathbf{c} = (100, 30, 0)$
  - "regular" classification cost for label 2: $\mathbf{c} = (1, 0, 1, 1)$
  - binary cost-sensitive classification cost for label -1: $\mathbf{c} = (0, C_{-1})$

multiclass, cost-vector based:
a very general setup

Cost-sensitive Classification Setup

Given

N examples, each (input $\mathbf{x}_n$, label y_n, cost $\mathbf{c}_n$) $\in \mathcal{X} \times \{1, 2, \ldots, K\} \times R^K$
—will assume $\mathbf{c}_n[y_n] = 0 = \min_{1 \le k \le K} \mathbf{c}_n[k]$

Goal

a classifier $g(\mathbf{x})$ that pays a small cost $\mathbf{c}[g(\mathbf{x})]$ on future unseen example $(\mathbf{x}, y, \mathbf{c})$

- will assume $\mathbf{c}[y] = 0 = \min_{1 \le k \le K} \mathbf{c}[k] = c_{\min}$
- **note: y not really needed in evaluation**

cost-sensitive classification:
can express any finite-loss supervised learning tasks

Our Contributions

decomposition	per-class	pair-wise	tournament	err. correcting
regular	OVA	OVO	FT	ECOC
cost-sensitive	**our work**	WAP	FT	SECOC

a theoretic and algorithmic study of multiclass cost-sensitive classification, which ...

- introduces a methodology to **reduce** cost-sensitive classification to **regression**
- couples the methodology with a novel regression loss for **strong theoretical support**
- leads to a promising SVM-based algorithm with **superior experimental results**

Key Idea: Cost Estimator

Goal

a classifier $g(\mathbf{x})$ that pays a small cost $\mathbf{c}[g(\mathbf{x})]$ on future unseen example $(\mathbf{x}, y, \mathbf{c})$

if every $\mathbf{c}[k]$ known

best $g^*(\mathbf{x}) = \underset{1 \le k \le K}{\operatorname{argmin}}\ \mathbf{c}[k]$

if $r_k(\mathbf{x}) \approx \mathbf{c}[k]$ well

approximately good $g_r(\mathbf{x}) = \underset{1 \le k \le K}{\operatorname{argmin}}\ r_k(\mathbf{x})$

how to get cost estimator r_k? **regression**

Cost Estimator by Regression

Given

N examples, each (input $\mathbf{x}_n$, label y_n, cost $\mathbf{c}_n$) $\in \mathcal{X} \times \{1, 2, \ldots, K\} \times R^K$

$$
\underbrace{\left\| \begin{array}{cc} \text{input} & \mathbf{c}_n[1] \\ \mathbf{x}_1 & 0, \\ \mathbf{x}_2 & 1, \\ \cdots & \\ \mathbf{x}_N & 6, \end{array} \right\|}_{r_1}
\underbrace{\left\| \begin{array}{cc} \text{input} & \mathbf{c}_n[2] \\ \mathbf{x}_1 & 2, \\ \mathbf{x}_2 & 3, \\ & \\ \mathbf{x}_N & 1, \end{array} \right\|}_{r_2}
\ldots
\underbrace{\left\| \begin{array}{cc} \text{input} & \mathbf{c}_n[K] \\ \mathbf{x}_1 & 1 \\ \mathbf{x}_2 & 5 \\ & \\ \mathbf{x}_N & 0 \end{array} \right\|}_{r_K}
$$

want: $r_k(\mathbf{x}) \approx \mathbf{c}[k]$ for all future $(\mathbf{x}, y, \mathbf{c})$ and k

good $r_k \Longrightarrow$ good g_r?

Absolute Loss Bound

$$g_r(\mathbf{x}) = \operatorname*{argmin}_{1 \le k \le K} r_k(\mathbf{x})$$

Theorem

For any set of regressors (cost estimators) $\{r_k\}_{k=1}^K$ and for any example $(\mathbf{x}, y, \mathbf{c})$ with $\mathbf{c}[y] = 0$,

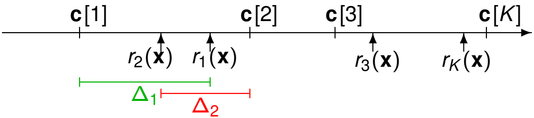
$$\mathbf{c}[g_r(\mathbf{x})] \le \sum_{k=1}^{K} \Big| r_k(\mathbf{x}) - \mathbf{c}[k] \Big|.$$

good $r_k \Longrightarrow$ good g_r? YES!

A Pictorial Proof

$$\mathbf{c}[g_r(\mathbf{x})] \le \sum_{k=1}^{K} \Big| r_k(\mathbf{x}) - \mathbf{c}[k] \Big|$$

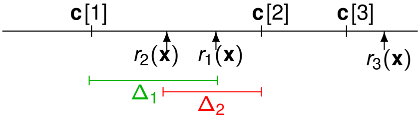
- assume $\mathbf{c}$ ordered and not degenerate:
  $y = 1$; $0 = \mathbf{c}[1] < \mathbf{c}[2] \le \cdots \le \mathbf{c}[K]$
- assume mis-prediction $g_r(\mathbf{x}) = 2$:
  $r_2(\mathbf{x}) = \min_{1 \le k \le K} r_k(\mathbf{x}) \le r_1(\mathbf{x})$



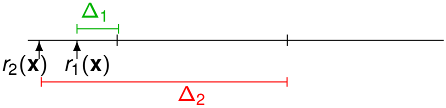
$$\mathbf{c}[2] - \underbrace{\mathbf{c}[1]}_{0} \le \Big|\Delta_1\Big| + \Big|\Delta_2\Big| \le \sum_{k=1}^{K} \Big| r_k(\mathbf{x}) - \mathbf{c}[k] \Big|$$

A Closer Look

let $\Delta_1 \equiv r_1(\mathbf{x}) - \mathbf{c}[1]$ and $\Delta_2 \equiv \mathbf{c}[2] - r_2(\mathbf{x})$



- $\Delta_1 \geq 0$ and $\Delta_2 \geq 0$: $\mathbf{c}[2] \leq \Delta_1 + \Delta_2$



- $\Delta_1 \leq 0$ and $\Delta_2 \geq 0$: $\mathbf{c}[2] \leq \Delta_2$

- $\Delta_1 \geq 0$ and $\Delta_2 \leq 0$: $\mathbf{c}[2] \leq \Delta_1$

$$\mathbf{c}[2] \leq \max(\Delta_1, 0) + \max(\Delta_2, 0) \leq |\Delta_1| + |\Delta_2|$$

Tighter Bound with One-sided Loss

Define **one-sided loss** $\xi_k \equiv \max(\Delta_k, 0)$,

$$\text{with} \quad \Delta_k \equiv \left(r_k(\mathbf{x}) - \mathbf{c}[k]\right) \text{ if } \mathbf{c}[k] = c_{\min}$$

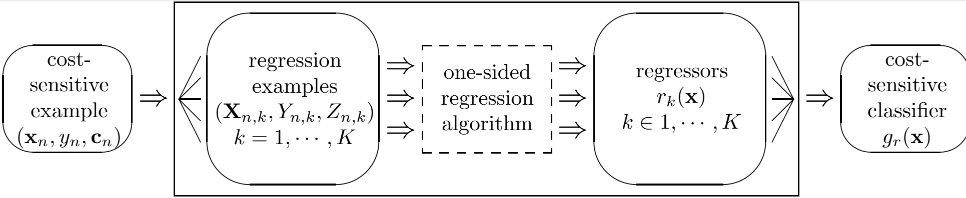
$$\Delta_k \equiv \left(\mathbf{c}[k] - r_k(\mathbf{x})\right) \text{ if } \mathbf{c}[k] \neq c_{\min}$$

Intuition: $\xi_k = 0$ encodes ...

- when $\mathbf{c}[k] = c_{\min}$: wish to have $r_k(\mathbf{x}) \leq \mathbf{c}[k]$
- when $\mathbf{c}[k] \neq c_{\min}$: wish to have $r_k(\mathbf{x}) \geq \mathbf{c}[k]$

$$\mathbf{c}[g_r(\mathbf{x})] \leq \underbrace{\sum_{k=1}^{K} \xi_k}_{\textbf{one-sided loss bound}} \leq \underbrace{\sum_{k=1}^{K} \left|\Delta_k\right|}_{\text{absolute loss bound}}$$

The Proposed Reduction Framework



1. transform cost-sensitive examples $(\mathbf{x}_n, y_n, \mathbf{c}_n)$ to regression examples $\left(\mathbf{X}_n^{(k)}, Y_n^{(k)}, Z_n^{(k)}\right) = \left(\mathbf{x}_n, \mathbf{c}_n[k], +/-\right)$
2. use a **one-sided regression algorithm** to get regressors $r_k(\mathbf{x})$
3. for each new input $\mathbf{x}$, predict its class using $g_r(\mathbf{x}) = \operatorname{argmin}_{1 \le k \le K} r_k(\mathbf{x})$

how to design a good OSR algorithm?

Support Vector Machinery for One-sided Regression

Given

$$(\mathbf{X}_{n,k}, Y_{n,k}, Z_{n,k}) = (\mathbf{x}_n, \mathbf{c}_n[k]\,, +/-)$$

Training Goal

all training $\xi_{n,k} = \max\left(\underbrace{Z_{n,k}\left(r_k(\mathbf{X}_{n,k}) - Y_{n,k}\right)}_{\Delta_{n,k}}, 0\right)$ small

OSR-SVM for cost-sensitive classification:

$$\min_{\mathbf{w}_k, b_k} \quad \frac{1}{2}\langle \mathbf{w}_k, \mathbf{w}_k \rangle + C\sum_{n=1}^{N} \xi_{n,k}$$

$$\text{to get} \quad r_k(\mathbf{X}) = \langle \mathbf{w}_k, \phi(\mathbf{X}) \rangle + b_k$$

One-sided Support Vector Regression

Standard Support Vector Regression

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle + C \sum_{n=1}^{N} (\xi_n + \xi_n^*)$$

$$\xi_n = \max\left(+ \left(r_k(\mathbf{X}_n) - Y_n - \epsilon \right), 0 \right)$$

$$\xi_n^* = \max\left(- \left(r_k(\mathbf{X}_n) - Y_n + \epsilon \right), 0 \right)$$

One-sided Support Vector Regression (for each k)

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle + C \sum_{n=1}^{N} \xi_n$$

$$\xi_n = \max\left(Z_n \cdot \left(r_k(\mathbf{X}_n) - Y_n \right), 0 \right)$$

OSR-SVM:

SVR + $(\epsilon = 0)$ + **(keep ξ_n or ξ_n^* by Z_n)**

OSR-SVM versus OVA-SVM: Formulations

OSR-SVM: $g_r(\mathbf{x}) = \text{argmin } r_k(\mathbf{X})$

$$\min_{\mathbf{w}_k, b_k} \quad \frac{1}{2}\langle \mathbf{w}_k, \mathbf{w}_k \rangle + C \sum_{n=1}^{N} \xi_{n,k}$$

$$\text{with} \quad r_k(\mathbf{X}) = \langle \mathbf{w}_k, \phi(\mathbf{X}) \rangle + b_k$$

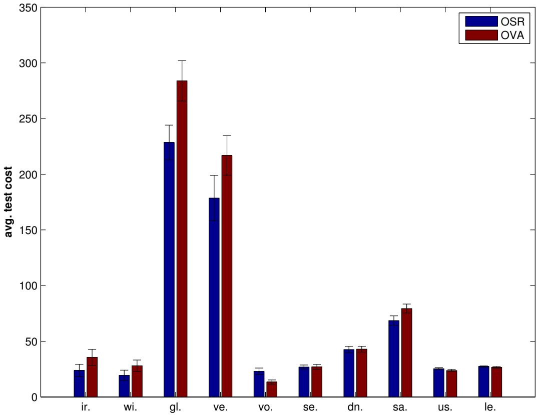
$$\xi_{n,k} = \max\left(Z_{n,k} \cdot \left(r_k(\mathbf{X}_{n,k}) - Y_{n,k}\right), 0\right)$$

OVA-SVM (-1 for correct class): $g_r(\mathbf{x}) = \text{argmin } r_k(\mathbf{X})$

$$\text{with} \quad \xi_{n,k} = \max\left(Z_{n,k} \cdot r_k(\mathbf{X}_{n,k}) + 1, 0\right)$$

OVA-SVM:
special case that replaces $Y_{n,k}$ (i.e. $\mathbf{c}_n[k]$) by $-Z_{n,k}$

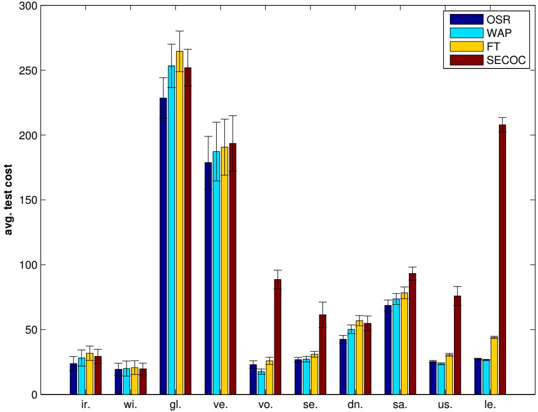
OSR-SVM versus OVA-SVM: Experiments



- OSR: a cost-sensitive extension of OVA
- OVA: cost-insensitive SVM

OSR often significantly better than OVA

OSR-SVM versus WAP/FT/SECOC-SVM



- OSR (per-class): $O(K)$ train/pred
- WAP (pair-wise): $O(K^2)$ train/pred
- FT (tournament): $O(K)$ train; $O(\log K)$ pred
- SECOC (err correct): big $O(K)$ train/pred

speed: FT > OSR > SECOC > WAP;
performance: OSR $\approx$ WAP > FT > SECOC

Conclusion

- **reduction to regression**:
  a simple way of designing cost-sensitive classification algorithms
- theoretical guarantee:
  absolute and **one-sided** bounds
- algorithmic use:
  a **novel and simple** algorithm OSR-SVM
- experimental performance of OSR-SVM:
  leading in SVM family

Thank you. Questions?