Cost-sensitive Multiclass Classification via Regression

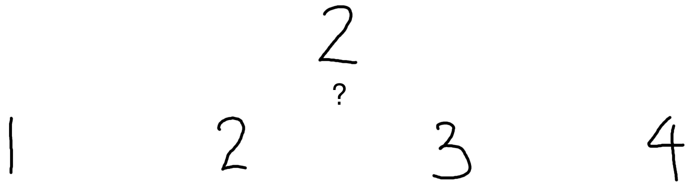
Hsuan-Tien Lin

Dept. of CSIE, NTU

Talk at NCU CSIE, 10/09/2013

Joint work with Han-Hsing Tu; parts appeared in ICML '10

Which Digit Did You Write?



one (1) two (2) three (3) four (4)

- a **classification** problem
  —grouping “pictures” into different “categories”

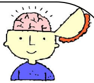
How can machines learn to classify?

Supervised Machine Learning

Parent

(picture, category) pairs

Kid's brain



→ good decision function

possibilities

Truth $f(\mathbf{x})$ + noise $e(\mathbf{x})$

examples (picture $\mathbf{x}_n$, category y_n)

learning algorithm

→ good decision function $g(\mathbf{x}) \approx f(\mathbf{x})$

learning model $\{g_\alpha(\mathbf{x})\}$

challenge:

see only $\{(\mathbf{x}_n, y_n)\}$ without knowing $f(\mathbf{x})$ or $e(\mathbf{x})$

$\overset{?}{\Longrightarrow}$ **generalize** to unseen $(\mathbf{x}, y)$ w.r.t. $f(\mathbf{x})$

Mis-prediction Costs ($g(\mathbf{x}) \approx f(\mathbf{x})$?)

?

- ZIP code recognition:
  1: wrong; 2: right; 3: wrong; 4: wrong
- check value recognition:
  1: one-dollar mistake; 2: no mistake;
  3: one-dollar mistake; 4: **two**-dollar mistake

different applications:
evaluate mis-predictions differently

ZIP Code Recognition

?

1: wrong; 2: right; 3: wrong; 4: right

- **regular** classification problem: only right or wrong
- wrong cost: 1; right cost: 0
- prediction error of g on some $(\mathbf{x}, y)$:

$$\text{classification cost} = [\![y \neq g(\mathbf{x})]\!]$$

regular classification:
well-studied, many good algorithms

Check Value Recognition

?

1: one-dollar mistake; 2: no mistake;
3: one-dollar mistake; 4: **two**-dollar mistake

- **cost-sensitive** classification problem:
  different costs for different mis-predictions
- e.g. prediction error of g on some $(\mathbf{x}, y)$:

$$\text{absolute cost} = |y - g(\mathbf{x})|$$

> cost-sensitive classification:
> **new**, need more research

What is the Status of the Patient?

?

H1N1-infected

cold-infected

healthy

- another **classification** problem
  —grouping “patients” into different “status”

Are all mis-prediction costs equal?

Patient Status Prediction

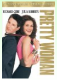
error measure = society cost

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

- H1N1 mis-predicted as healthy: **very high cost**
- cold mis-predicted as healthy: high cost
- cold correctly predicted as cold: no cost

human doctors consider costs of decision;
can computer-aided diagnosis do the same?

What is the Type of the Movie?

			
?	romance	fiction	terror

customer 1 who hates terror but likes romance

error measure = non-satisfaction

predicted / actual	romance	fiction	terror
romance	0	5	100

customer 2 who likes terror and romance

predicted / actual	romance	fiction	terror
romance	0	5	3

different customers:

evaluate mis-predictions differently

Cost-sensitive Classification Tasks

movie classification with non-satisfaction

actual \ predicted	romance	fiction	terror
customer 1, romance	0	5	100
customer 2, romance	0	5	3

patient diagnosis with society cost

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

check digit recognition with absolute cost

$$\mathcal{C}(y, g(\mathbf{x})) = |g(\mathbf{x}) - y|$$

Cost Vector

cost vector **c**: a row of cost components

- customer 1 on a romance movie: $\mathbf{c} = (0, 5, 100)$
- an H1N1 patient: $\mathbf{c} = (0, 1000, 100000)$
- absolute cost for digit 2: $\mathbf{c} = (1, 0, 1, 2)$
- "regular" classification cost for label 2: $\mathbf{c}_c^{(2)} = (1, 0, 1, 1)$

regular classification:
special case of cost-sensitive classification

Cost-sensitive Classification Setup

Given

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

- $K = 2$: binary; $K > 2$: **multiclass**
- 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 $\mathbf{c}[y] = 0 = c_{\min} = \min_{1 \le k \le K} \mathbf{c}[k]$
- note: y not really needed in evaluation

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

Our Contribution

	binary	multiclass
regular	well-studied	well-studied
cost-sensitive	known (Zadrozny, 2003)	ongoing (our work, among others)

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

- introduces a methodology to reduce cost-sensitive classification to **regression**
- provides **strong theoretical support** for the methodology
- leads to a promising algorithm with **superior experimental results**

will describe the methodology and an algorithm

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

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

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 Per-class Regression

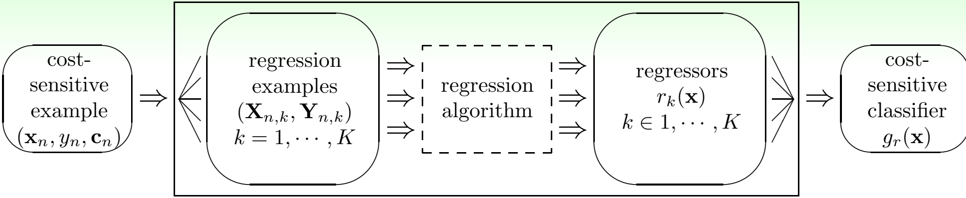
Given

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

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

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

The Reduction Framework



1. transform cost-sensitive examples $(\mathbf{x}_n, y_n, \mathbf{c}_n)$ to regression examples $\left(\mathbf{X}_{n,k}, Y_{n,k}\right) = \left(\mathbf{x}_n, \mathbf{c}_n[k]\right)$
2. use your favorite algorithm on the regression examples and get regressors $r_k(\mathbf{x})$
3. for each new input $\mathbf{x}$, predict its class using $g_r(\mathbf{x}) = \underset{1 \le k \le K}{\text{argmin}}\ r_k(\mathbf{x})$

the reduction-to-regression framework:
systematic & easy to implement

Theoretical Guarantees (1/2)

$$g_r(\mathbf{x}) = \underset{1 \le k \le K}{\text{argmin}} \; r_k(\mathbf{x})$$

Theorem (Absolute Loss Bound)

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

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

low-cost classifier $\Longleftarrow$ accurate regressor

Theoretical Guarantees (2/2)

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

Theorem (Squared Loss Bound)

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

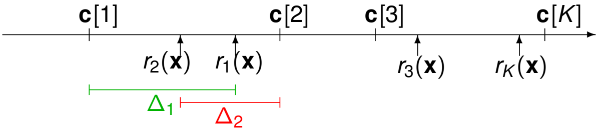
$$\mathbf{c}[g_r(\mathbf{x})] \le \sqrt{2 \sum_{k=1}^{K} \left(r_k(\mathbf{x}) - \mathbf{c}[k]\right)^2}.$$

applies to common **least-square regression**

A Pictorial Proof

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

- 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 $g_r(\mathbf{x}) = 2$:
  $r_2(\mathbf{x}) = \min_{1 \le k \le K} r_k(\mathbf{x}) \le r_1(\mathbf{x})$



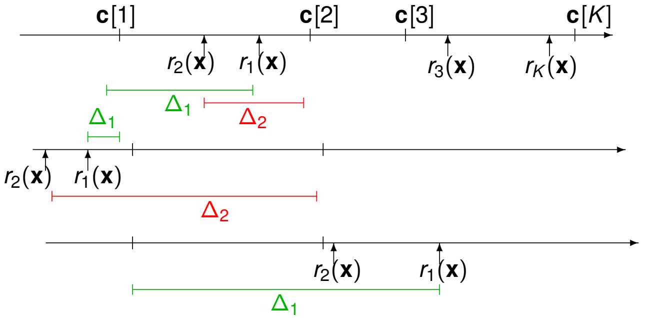
$$\mathbf{c}[2] - \underbrace{\mathbf{c}[1]}_{0} \le |\Delta_1| + |\Delta_2| \le \sum_{k=1}^{K} \left| r_k(\mathbf{x}) - \mathbf{c}[k] \right|$$

An Even Closer Look

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

1. $\Delta_1 \geq 0$ and $\Delta_2 \geq 0$: $\mathbf{c}[2] \leq \Delta_1 + \Delta_2$
2. $\Delta_1 \leq 0$ and $\Delta_2 \geq 0$: $\mathbf{c}[2] \leq \Delta_2$
3. $\Delta_1 \geq 0$ 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** $\xi_k \equiv \max(\Delta_k, 0)$

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

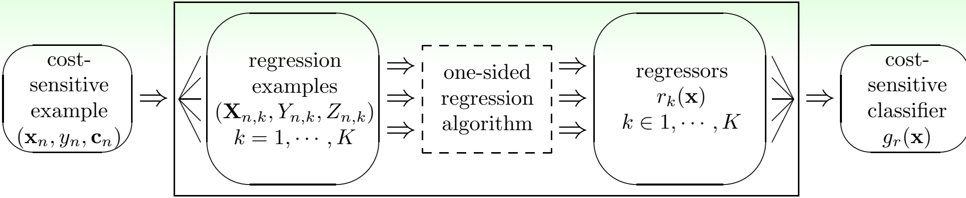
Intuition

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

—**both wishes same as** $\Delta_k \leq 0$ **and hence** $\xi_k = 0$

One-sided Loss Bound:
$$\mathbf{c}[g_r(\mathbf{x})] \leq \sum_{k=1}^{K} \xi_k \leq \sum_{k=1}^{K} \Big|\Delta_k\Big|$$

The Improved Reduction Framework



1. transform cost-sensitive examples $(\mathbf{x}_n, y_n, \mathbf{c}_n)$ to regression examples $\left(\mathbf{X}_n^{(k)}, Y_n^{(k)}, Z_n^{(k)}\right) = \left(\mathbf{x}_n, \mathbf{c}_n[k], 2\left[\!\left[\mathbf{c}_n[k] = \mathbf{c}_n[y_n]\right]\!\right] - 1\right)$
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 $g_r(\mathbf{x}) = \operatorname{argmin}_{1 \le k \le K} r_k(\mathbf{x})$

the reduction-to-OSR framework:
need a good OSR algorithm

Regularized One-sided Hyper-linear Regression

Given

$$\left(\mathbf{X}_{n,k}, Y_{n,k}, Z_{n,k}\right) = \left(\mathbf{x}_n, \mathbf{c}_n[k], 2\left[\!\left[\mathbf{c}_n[k] = \mathbf{c}_n[y_n]\right]\!\right] - 1\right)$$

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

—will drop k

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

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

One-sided Support Vector Regression

Regularized One-sided Hyper-linear Regression

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

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

Standard Support Vector Regression

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

$$\xi_n = \max\left(+1 \cdot (r_k(\mathbf{X}_n) - Y_n - \epsilon), 0\right)$$

$$\xi_n^* = \max\left(-1 \cdot (r_k(\mathbf{X}_n) - Y_n + \epsilon), 0\right)$$

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

OSR versus Other Reductions

OSR: K regressors

How unlikely (costly) does the example belong to class k?

Filter Tree (FT): $K - 1$ binary classifiers

Is the lowest cost within labels $\{1, 4\}$ or $\{2, 3\}$?
Is the lowest cost within label $\{1\}$ or $\{4\}$?
Is the lowest cost within label $\{2\}$ or $\{3\}$?

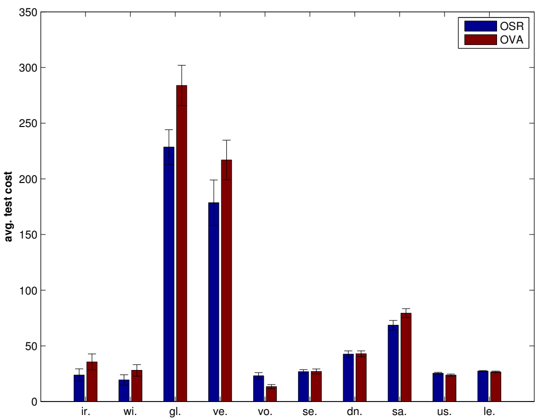
Weighted All Pairs (WAP): $\frac{K(K-1)}{2}$ binary classifiers

is $\mathbf{c}[1]$ or $\mathbf{c}[4]$ lower?

Sensitive Error Correcting Output Code (SECOC): $(T \cdot K)$ bin. cla.

is $\mathbf{c}[1] + \mathbf{c}[3] + \mathbf{c}[4]$ greater than some θ?

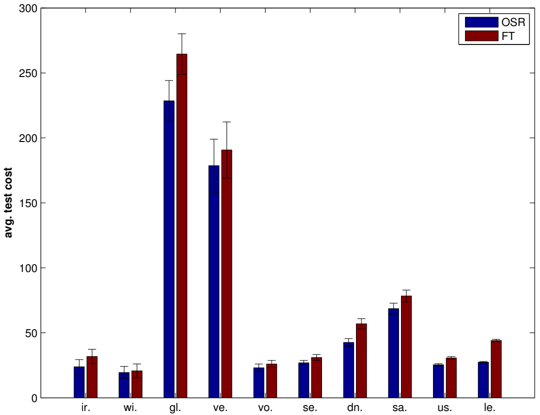
Experiment: OSR-SVM versus OVA-SVM



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

OSR often significantly better than OVA

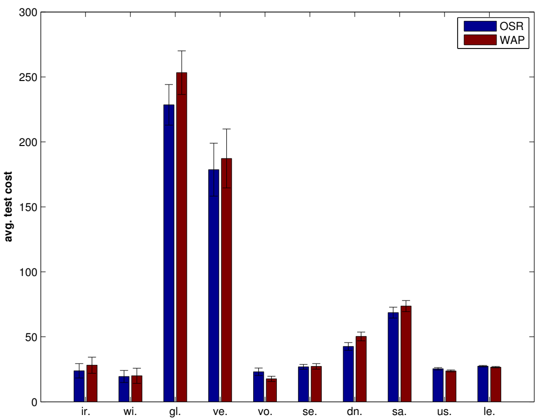
Experiment: OSR versus FT



- OSR (per-class): $O(K)$ training, $O(K)$ prediction
- FT (tournament): $O(K)$ training, $O(\log_2 K)$ prediction

FT faster, but OSR better performed

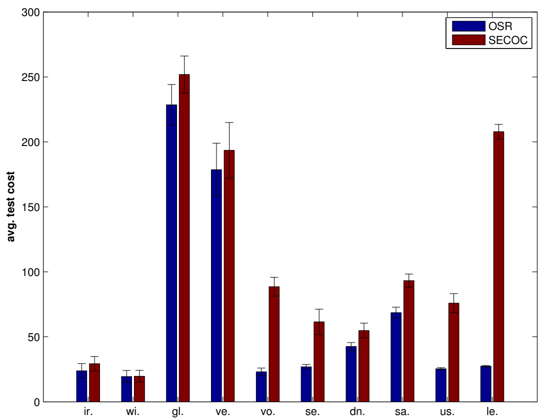
Experiment: OSR versus WAP



- OSR (per-class): $O(K)$ training, $O(K)$ prediction
- WAP (pairwise): $O(K^2)$ training, $O(K^2)$ prediction

OSR faster and comparable performance

Experiment: OSR versus SECOC



- OSR (per-class): $O(K)$ training, $O(K)$ prediction
- SECOC (error-correcting): big $O(K)$ training, big $O(K)$ prediction

OSR faster and much better performance

Conclusion

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

more algorithm and application opportunities

Acknowledgments

- Profs. Chih-Jen Lin, Yuh-Jyh Lee, Shou-de Lin for suggestions
- Prof. Chia-Hui Chang for talk invitation
- Computational Learning Lab @ NTU for discussions

Thank you. Questions?