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

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>Cold</th>
<th>Healthy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>0</td>
<td>$C_{-1}$</td>
<td></td>
</tr>
<tr>
<td>Healthy</td>
<td>$C_{1}$</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

binary, cost-matrix based
Multiclass Cost-sensitive Classification

- **Error Measure**: Society Cost
- **Table**:

<table>
<thead>
<tr>
<th></th>
<th>H1N1</th>
<th>cold</th>
<th>healthy</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual: H1N1</td>
<td>0</td>
<td>1000</td>
<td>100000</td>
</tr>
<tr>
<td>actual: cold</td>
<td>100</td>
<td>0</td>
<td>3000</td>
</tr>
<tr>
<td>actual: healthy</td>
<td>100</td>
<td>30</td>
<td>0</td>
</tr>
</tbody>
</table>

- Human doctors consider costs of decision
- Want computer-aided diagnosis to behave similarly

**Multiclass**, cost-matrix based
Cost-sensitive Classification

From Cost Matrix to Cost Vector

with actual underlying status

<table>
<thead>
<tr>
<th>prediction</th>
<th>H1N1</th>
<th>cold</th>
<th>healthy</th>
</tr>
</thead>
<tbody>
<tr>
<td>society cost</td>
<td>0</td>
<td>1000</td>
<td>100000</td>
</tr>
</tbody>
</table>

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 \((\text{input } x_n, \text{label } y_n, \text{cost } c_n) \in \mathcal{X} \times \{1, 2, \ldots, K\} \times \mathbb{R}^K\)

— will assume \( c_n[y_n] = 0 = \min_{1 \leq k \leq K} c_n[k] \)

**Goal**

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

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

**cost-sensitive classification:**

can express any finite-loss supervised learning tasks
Our Contributions

<table>
<thead>
<tr>
<th>decomposition</th>
<th>per-class</th>
<th>pair-wise</th>
<th>tournament</th>
<th>err. correcting</th>
</tr>
</thead>
<tbody>
<tr>
<td>regular</td>
<td>OVA</td>
<td>OVO</td>
<td>FT</td>
<td>ECOC</td>
</tr>
<tr>
<td>cost-sensitive</td>
<td>our work</td>
<td>WAP</td>
<td>FT</td>
<td>SECOC</td>
</tr>
</tbody>
</table>

*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 \( g(x) \) that pays a small cost \( c[g(x)] \) on future unseen example \((x, y, c)\)

If every \( c[k] \) known

\[
\text{best } g^*(x) = \arg\min_{1 \leq k \leq K} c[k]
\]

If \( r_k(x) \approx c[k] \) well

\[
\text{approximately good } g_r(x) = \arg\min_{1 \leq k \leq K} r_k(x)
\]

How to get cost estimator \( r_k \)? regression
Given

\( N \) examples, each \((\text{input } x_n, \text{label } y_n, \text{cost } c_n) \in \mathcal{X} \times \{1, 2, \ldots, K\} \times \mathbb{R}^K\)

<table>
<thead>
<tr>
<th>input</th>
<th>( c_n[1] )</th>
<th>input</th>
<th>( c_n[2] )</th>
<th>...</th>
<th>input</th>
<th>( c_n[K] )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>0,</td>
<td>( x_1 )</td>
<td>2,</td>
<td>...</td>
<td>( x_1 )</td>
<td>1</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>1,</td>
<td>( x_2 )</td>
<td>3,</td>
<td></td>
<td>( x_2 )</td>
<td>5</td>
</tr>
<tr>
<td>( \ldots )</td>
<td></td>
<td>( x_N )</td>
<td>1,</td>
<td></td>
<td>( x_N )</td>
<td>0</td>
</tr>
<tr>
<td>( r_1 )</td>
<td></td>
<td>( r_2 )</td>
<td></td>
<td></td>
<td>( r_K )</td>
<td></td>
</tr>
</tbody>
</table>

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

\[ \text{good } r_k \implies \text{good } gr? \]
Absolute Loss Bound

Design and Analysis

Theorem

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

\[
c[gr(x)] \leq \sum_{k=1}^{K} |r_k(x) - c[k]|.
\]

**good** \( r_k \) \( \iff \) **good** \( gr \) ? **YES!**
A Pictorial Proof

\[ c[gr(x)] \leq \sum_{k=1}^{K} |r_k(x) - c[k]| \]

- Assume \( c \) ordered and not degenerate:
  \( y = 1; 0 = c[1] < c[2] \leq \cdots \leq c[K] \)
- Assume mis-prediction \( gr(x) = 2 \):
  \( r_2(x) = \min_{1 \leq k \leq K} r_k(x) \leq r_1(x) \)

\[ c[2] - c[1] \leq |\Delta_1| + |\Delta_2| \leq \sum_{k=1}^{K} |r_k(x) - c[k]| \]
let $\Delta_1 \equiv r_1(x) - c[1]$ and $\Delta_2 \equiv c[2] - r_2(x)$

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

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

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

$c[2] \leq \max(\Delta_1, 0) + \max(\Delta_2, 0) \leq |\Delta_1| + |\Delta_2|$
Define **one-sided loss** \( \xi_k \equiv \max(\Delta_k, 0) \),

with

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

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

Intuition: \( \xi_k = 0 \) encodes ...

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

\[
c[g_r(x)] \leq \sum_{k=1}^{K} \xi_k \leq \sum_{k=1}^{K} |\Delta_k| \quad \text{one-sided loss bound}
\]

\[
\text{absolute loss bound}
\]
1. transform cost-sensitive examples \((x_n, y_n, c_n)\) to regression examples \((X_{n,k}^{(k)}, Y_{n,k}^{(k)}, Z_{n,k}^{(k)}) = (x_n, c_n[k], +/-)\)

2. use a **one-sided regression algorithm** to get regressors \(r_k(x)\)

3. for each new input \(x\), predict its class using \(g_r(x) = \arg\min_{1 \leq k \leq K} r_k(x)\)

**how to design a good OSR algorithm?**
**The Proposed Algorithm**

**Support Vector Machinery for One-sided Regression**

**Given**

\[ (X_{n,k}, Y_{n,k}, Z_{n,k}) = (x_n, c_n[k], +/-) \]

**Training Goal**

all training \( \xi_{n,k} = \max \left( \frac{Z_{n,k} (r_k(X_{n,k}) - Y_{n,k})}{\Delta_{n,k}}, 0 \right) \) small

**OSR-SVM for cost-sensitive classification:**

\[
\min_{w_k, b_k} \frac{1}{2} \langle w_k, w_k \rangle + C \sum_{n=1}^{N} \xi_{n,k} \\
to~get~ r_k(X) = \langle w_k, \phi(X) \rangle + b_k
\]
The Proposed Algorithm

One-sided Support Vector Regression

Standard Support Vector Regression

\[
\min_{w, b} \frac{1}{2} \langle w, w \rangle + C \sum_{n=1}^{N} (\xi_n + \xi^*_n)
\]

\[
\xi_n = \max (r_k(X_n) - Y_n - \epsilon, 0)
\]

\[
\xi^*_n = \max (-(r_k(X_n) - Y_n + \epsilon), 0)
\]

One-sided Support Vector Regression (for each \(k\))

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

\[
\xi_n = \max (Z_n \cdot (r_k(X_n) - Y_n), 0)
\]

OSR-SVM:

\[
SVR + (\epsilon = 0) + (\text{keep } \xi_n \text{ or } \xi^*_n \text{ by } Z_n)
\]
OSR-SVM: \( g_r(x) = \text{argmin } r_k(X) \)

\[
\min_{w_k,b_k} \frac{1}{2} \langle w_k, w_k \rangle + C \sum_{n=1}^{N} \xi_{n,k} \\
\text{with } r_k(X) = \langle w_k, \phi(X) \rangle + b_k \\
\xi_{n,k} = \max (Z_{n,k} \cdot (r_k(X_{n,k}) - Y_{n,k}) , 0)
\]

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

\[
\text{with } \xi_{n,k} = \max (Z_{n,k} \cdot r_k(X_{n,k}) + 1 , 0)
\]

OVA-SVM: **special case** that replaces \( Y_{n,k} \) (i.e. \( c_n[k] \)) 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
Experiments

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?