Cost-Sensitive Classification: Algorithms and Advances

Hsuan-Tien Lin
htlin@csie.ntu.edu.tw

Department of Computer Science & Information Engineering
National Taiwan University

Tutorial for ACML @ Canberra, Australia
November 13, 2013
More about Me

- co-leader of KDDCup world champion teams at NTU: 2010–2013
- research on multi-label classification, ranking, active learning, etc.
- research on cost-sensitive classification: 2007–Present
- Secretary General, Taiwanese Association for Artificial Intelligence
- instructor of Mandarin-teaching MOOC of Machine Learning on NTU-Coursera: 2013.11–
  https://www.coursera.org/course/ntumlone
Outline

Cost-Sensitive Binary Classification

Bayesian Perspective of Cost-Sensitive Binary Classification
Non-Bayesian Perspective of Cost-Sensitive Binary Classification

Cost-Sensitive Multiclass Classification
Bayesian Perspective of Cost-Sensitive Multiclass Classification

Cost-Sensitive Classification by Reweighting and Relabeling
Cost-Sensitive Classification by Binary Classification
Cost-Sensitive Classification by Regression

Cost-and-Error-Sensitive Classification with Bioinformatics Application
Cost-Sensitive Ordinal Ranking with Information Retrieval Application

Summary
Is This Your Fingerprint?

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

C’mon, we know about binary classification all too well! :-)

Hsuan-Tien Lin (NTU CSIE)

Cost-Sensitive Classification: Algorithms and Advances
Supervised Machine Learning

(parent) -> (picture, category) pairs

truth $f(x) + \text{noise } e(x)$

effects $x_n$, category $y_n$

(kid's brain) -> good decision function

possibilities

learning algorithm -> good decision function $g(x) \approx f(x)$

learning model $\{h(x)\}$

how to evaluate whether $g(x) \approx f(x)$?
**Performance Evaluation**

**Fingerprint Verification**

Example/figure borrowed from Amazon ML best-seller textbook

"Learning from Data" (Abu-Mostafa, Magdon-Ismail, 2013)

Two types of error: **false accept** and **false reject**

<table>
<thead>
<tr>
<th>$f$</th>
<th>$g$</th>
<th>$g$</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1</td>
<td>+1</td>
<td>+1 0</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1 1</td>
</tr>
</tbody>
</table>

Simplest choice: penalizes both types **equally** and calculate **average** penalties
Fingerprint Verification for Supermarket

Fingerprint Verification

two types of error: **false accept** and **false reject**

<table>
<thead>
<tr>
<th>$f$</th>
<th>$g$</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1</td>
<td>+1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

- **false accept**: give a minor discount, intruder left fingerprint :-(
- **false reject**: very unhappy customer, lose future business

- supermarket: fingerprint for discount
Fingerprint Verification for CIA

two types of error: **false accept** and **false reject**

<table>
<thead>
<tr>
<th></th>
<th>+1</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+1</td>
<td>no error</td>
<td>false reject</td>
</tr>
<tr>
<td>-1</td>
<td>false accept</td>
<td>no error</td>
</tr>
</tbody>
</table>

- CIA: fingerprint for entrance
- **false accept**: very serious consequences!
- **false reject**: unhappy employee, but so what? :-)

\[ f \quad \rightarrow \quad \begin{cases} +1 & \text{you} \\ -1 & \text{intruder} \end{cases} \]
Cost-Sensitive Binary Classification

Regular Binary Classification

penalizes both types equally

<table>
<thead>
<tr>
<th></th>
<th>$h(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
</tr>
</tbody>
</table>

in-sample error for any hypothesis $h$

$$E_{in}(h) = \frac{1}{N} \sum \left[ y_n \neq h(x_n) \right]$$

out-of-sample error for any hypothesis $h$

$$E_{out}(h) = \mathcal{E} \sum \left[ y \neq h(x) \right]$$

regular binary classification:
well-studied in machine learning
—ya, we know! :-)

Hsuan-Tien Lin  (NTU CSIE)
Supermarket Cost (Error, Loss, ...) Matrix

<table>
<thead>
<tr>
<th></th>
<th>$h(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$+1$</td>
<td>0</td>
</tr>
<tr>
<td>$-1$</td>
<td>1</td>
</tr>
</tbody>
</table>

in-sample

$$E_{in}(h) = \frac{1}{N} \sum_{n=1}^{N} \left\{ \begin{array}{ll} 10 & \text{if } y_n = +1 \\ 1 & \text{if } y_n = -1 \end{array} \right\} \cdot \mathbb{I}[y_n \neq h(x_n)]$$

out-of-sample

$$E_{out}(h) = \frac{\mathcal{E}}{(x,y)} \left\{ \begin{array}{ll} 10 & \text{if } y = +1 \\ 1 & \text{if } y = -1 \end{array} \right\} \cdot \mathbb{I}[y \neq h(x)]$$

class-weighted cost-sensitive binary classification: different ‘weight’ for different $y$
Setup: Class-Weighted Cost-Sensitive Binary Classification

**Given**

- \( N \) examples, each \((input \, x_n, \, label \, y_n) \in \mathcal{X} \times \{-1, +1\}\)
- and weights \( w_+ \), \( w_- \)
- representing the two entries of the cost matrix

\[
\begin{array}{c|cc}
\text{y} & +1 & -1 \\
\hline
h(x) & 0 & w_+ \\
-1 & w_- & 0 \\
\end{array}
\]

**Goal**

- a classifier \( g(x) \) that pays a small cost \( w_y \) \([y \neq g(x)]\)
- on future **unseen** example \((x, y)\), i.e., achieves low \( E_{out}(g) \)

regular classification: \( w_+ = w_- \) (= 1)
Fingerprint Verification

two types of error: false accept and false reject

- supermarket: fingerprint for discount
- big customer: really don’t want to lose her/his business
- usual customer: don’t want to lose business, but not so serious
Example-Weighted Cost-Sensitive Binary Classification

Supermarket Cost Vectors (Rows)

<table>
<thead>
<tr>
<th></th>
<th>+1</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>big</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>usual</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>intruder</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

in-sample

\[
E_{in}(h) = \frac{1}{N} \sum_{n=1}^{N} w_n \cdot [y_n \neq h(x_n)]
\]

out-of-sample

\[
E_{out}(h) = \mathcal{E}_{(x,y,w)} w \cdot [y \neq h(x)]
\]

element-weighted cost-sensitive binary classification: different \(w\) for different \((x, y)\)
—seen this in AdaBoost? :-)
Setup: Example-Weighted Cost-Sensitive Binary Classification

**Given**

\( N \) examples, each \((input \ x_n, label \ y_n) \in \mathcal{X} \times \{-1, +1\}\) and weight \(w_n \in \mathbb{R}^+\)

**Goal**

a classifier \(g(x)\) that

\[
pays \quad \text{a small cost} \quad w\|y \neq g(x)\|
\]

on future **unseen** example \((x, y, w)\), i.e., achieves low \(E_{out}(g)\)

regular \(\subset\) class-weighted \(\subset\) example-weighted
Outline

Cost-Sensitive Binary Classification

Bayesian Perspective of Cost-Sensitive Binary Classification

Non-Bayesian Perspective of Cost-Sensitive Binary Classification

Cost-Sensitive Multiclass Classification

Bayesian Perspective of Cost-Sensitive Multiclass Classification

Cost-Sensitive Classification by Reweighting and Relabeling

Cost-Sensitive Classification by Binary Classification

Cost-Sensitive Classification by Regression

Cost-and-Error-Sensitive Classification with Bioinformatics Application

Cost-Sensitive Ordinal Ranking with Information Retrieval Application

Summary
Key Idea: Conditional Probability Estimator

Goal (Class-Weighted Setup)

A classifier \( g(x) \) that pays a small cost \( w_y \) \([y \neq g(x)]\) on future \textbf{unseen} example \((x, y)\)

- expected error for predicting +1 on \( x \): \( w_+ P(+1|x) \)
- expected error for predicting -1 on \( x \): \( w_- P(-1|x) \)

If \( P(y|x) \) known

Bayes optimal \( g^*(x) = \)

\[
\text{sign}\left(w_+ P(+1|x) - w_- P(-1|x)\right)
\]

If \( p(x) \approx P(+1|x) \) well

approximately good \( g_p(x) = \)

\[
\text{sign}\left(w_+ p(x) - w_- (1 - p(x))\right)
\]

How to get conditional probability estimator \( p \)?

\textbf{logistic regression, Naïve Bayes, …}
if $p(x) \approx P(+1|x)$ well

approximately good $g_p(x) = \text{sign} \left( w_+ p(x) - w_- (1 - p(x)) \right)$

that is (Elkan, 2001),

$g_p(x) = +1 \text{ iff } w_+ p(x) - w_- (1 - p(x)) > 0$

iff $p(x) > \frac{w_-}{w_+ + w_-} : \frac{1}{11}$ for supermarket; $\frac{100}{101}$ for CIA

Approximate Bayes-Optimal Decision (ABOD) Approach

1. use your favorite algorithm on $\{(x_n, y_n)\}$ to get $p(x) \approx P(+1|x)$
2. for each new input $x$, predict its class using $g_p(x) = \text{sign}(p(x) - \frac{w_-}{w_+ + w_-})$

‘simplest’ approach:
probability estimate + threshold changing
1. use your favorite algorithm on \( \{(x_n, y_n)\} \) to get \( p(x) \approx P(+1|x) \)

2. for each new input \( x \), predict its class using

\[
g_p(x) = \text{sign}(p(x) - \frac{w_-}{w_+ + w_-})
\]

LogReg $\rightarrow$ g

<table>
<thead>
<tr>
<th>y</th>
<th>+1</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

regular  supermarket
Pros and Cons of ABOD

**Pros**

- **optimal**: if good probability estimate: $p(x)$ really close to $P(+1|x)$
- **simple**: training (probability estimate) unchanged, and prediction (threshold) changed only a little

**Cons**

- ‘difficult’: good probability estimate often more difficult than good binary classification
- ‘restricted’: only applicable to class-weighted setup —need ‘full picture’ of cost matrix

approach for the example-weighted setup?
Non-Bayesian Perspective of Cost-Sensitive Binary Classification

Outline

Cost-Sensitive Binary Classification

Bayesian Perspective of Cost-Sensitive Binary Classification

Non-Bayesian Perspective of Cost-Sensitive Binary Classification

Cost-Sensitive Multiclass Classification

Bayesian Perspective of Cost-Sensitive Multiclass Classification

Cost-Sensitive Classification by Reweighting and Relabeling

Cost-Sensitive Classification by Binary Classification

Cost-Sensitive Classification by Regression

Cost-and-Error-Sensitive Classification with Bioinformatics Application

Cost-Sensitive Ordinal Ranking with Information Retrieval Application

Summary
Key Idea: Example Weight = Copying

Goal

A classifier $g(x)$ that pays a small cost $w \mathbb{1}[y \neq g(x)]$

On future \textbf{unseen} example $(x, y, w)$

On one $(x, y)$

Wrong prediction charged by $w$

On $w$ copies of $(x, y)$

Wrong prediction charged by 1 —regular classification

How to copy? \textbf{over-sampling}
Example-Weighted Classification by Over-Sampling

**copy each** \((x_n, y_n)\) **for** \(w_n\) **times**

<table>
<thead>
<tr>
<th>(y)</th>
<th>big</th>
<th>usual</th>
<th>intruder</th>
</tr>
</thead>
<tbody>
<tr>
<td>(h(x))</td>
<td>+1</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>big</td>
<td>0</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>usual</td>
<td>0</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>intruder</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

\((x_1, -1, 1)\)
\((x_2, +1, 10)\)
\((x_3, +1, 100)\)
\((x_4, +1, 10)\)
\((x_5, -1, 1)\)

**equivalent problem**

<table>
<thead>
<tr>
<th>(y)</th>
<th>big</th>
<th>usual</th>
<th>intruder</th>
</tr>
</thead>
<tbody>
<tr>
<td>(h(x))</td>
<td>+1</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>big</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>usual</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>intruder</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

\((x_1, -1)\)
\((x_2, +1), \ldots, (x_2, +1)\)
\((x_3, +1), \ldots, (x_3, +1), \ldots, (x_3, +1)\)
\((x_4, +1), \ldots, (x_4, +1)\)
\((x_5, -1)\)

**how to learn a good \(g\) for RHS?**

**SVM, NNet, \ldots**
Cost-Proportionate Example Weighting

Cost-Proportionate Example Weighting (CPEW) Approach

1. Effectively transform \( \{(x_n, y_n, w_n)\} \) to \( \{(x_m, y_m)\} \) such that the 'copies' of \((x_n, y_n)\) in \( \{(x_m, y_m)\} \) is proportional to \(w_n\)
   - over/under-sampling with normalized \(w_n\) (Elkan, 2001)
   - under-sampling by rejection (Zadrozny, 2003)
   - modify existing algorithms equivalently (Zadrozny, 2003)

2. Use your favorite algorithm on \( \{(x_m, y_m)\} \) to get binary classifier \(g(x)\)

3. For each new input \(x\), predict its class using \(g(x)\)

Simple and general:
very popular for cost-sensitive binary classification
CPEW by Modification

1. effectively transform \( \{(x_n, y_n, w_n)\} \) to \( \{(x_m, y_m)\} \) such that the ‘copies’ of \( (x_n, y_n) \) in \( \{(x_m, y_m)\} \) is proportional to \( w_n \)
   - modify existing algorithms equivalently (Zadrozny, 2003)

2. use your favorite algorithm on \( \{(x_m, y_m)\} \) to get binary classifier \( g(x) \)

3. for each new input \( x \), predict its class using \( g(x) \)

**Regular Linear SVM**

\[
\min_{w, b} \quad \frac{1}{2} \langle w, w \rangle + \sum_{n=1}^{N} C \xi_n \\
\xi_n = \max (1 - y_n(\langle w, x_n \rangle + b), 0)
\]

**Modified Linear SVM**

\[
\min_{w, b} \quad \frac{1}{2} \langle w, w \rangle + \sum_{n=1}^{N} C \cdot w_n \cdot \xi_n \\
\xi_n = \max (1 - y_n(\langle w, x_n \rangle + b), 0)
\]
1. Effectively transform \( \{(x_n, y_n, w_n)\} \) to \( \{(x_m, y_m)\} \) by modifying existing algorithms equivalently (Zadrozny, 2003).
2. Use your favorite algorithm on \( \{(x_m, y_m)\} \) to get \( g(x) \).
3. For each new input \( x \), predict its class using \( g(x) \).
Non-Bayesian Perspective of Cost-Sensitive Binary Classification

CPEW by Rejection Sampling

COSTING Algorithm (Zadrozny, 2003)

1. Effectively transform \( \{(x_n, y_n, w_n)\} \) to \( \{(x_m, y_m)\} \) such that the ‘copies’ of \((x_n, y_n)\) in \( \{(x_m, y_m)\} \) is proportional to \( w_n \)
   - under-sampling by rejection (Zadrozny, 2003)

2. Use your favorite algorithm on \( \{(x_m, y_m)\} \) to get binary classifier \( g(x) \)

3. Repeat 1 and 2 to get multiple \( g \) and aggregate them

4. For each new input \( x \), predict its class using aggregated \( g(x) \)

Commonly used when your favorite algorithm is a black box rather than a white box
Biased Personal Favorites

- CPEW by Modification if possible
- COSTING: fast training and stable performance
- ABOD if in the mood for Bayesian :-(
Cost-Sensitive Multiclass Classification

Outline

Cost-Sensitive Binary Classification

Bayesian Perspective of Cost-Sensitive Binary Classification

Non-Bayesian Perspective of Cost-Sensitive Binary Classification

Cost-Sensitive Multiclass Classification

Bayesian Perspective of Cost-Sensitive Multiclass Classification

Cost-Sensitive Classification by Reweighting and Relabeling

Cost-Sensitive Classification by Binary Classification

Cost-Sensitive Classification by Regression

Cost-and-Error-Sensitive Classification with Bioinformatics Application

Cost-Sensitive Ordinal Ranking with Information Retrieval Application

Summary
Which Digit Did You Write?

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

C’mon, we know about
multiclass classification all too well! :-)
Cost-Sensitive Multiclass Classification

Performance Evaluation \((g(x) \approx f(x)?)\)

- 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
- evaluation by formation similarity:
  1: not very similar; 2: very similar;
  3: somewhat similar; 4: a silly prediction

different applications:
**evaluate mis-predictions differently**
ZIP Code Recognition

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

- **regular** multiclass classification: only right or wrong
- wrong cost: 1; right cost: 0
- prediction error of $h$ on some $(x, y)$:

  $$\text{classification cost} = \left\lceil y \neq h(x) \right\rceil$$

— as discussed in regular binary classification

regular multiclass classification: **well-studied**, many good algorithms
Check Value Recognition

2

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

- cost-sensitive multiclass classification: different costs for different mis-predictions
- e.g. prediction error of $h$ on some $(x, y)$:

  \[
  \text{absolute cost} = |y - h(x)|
  \]

next: cost-sensitive multiclass classification
What is the Status of the Patient?

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

**H1N1-infected**  **cold-infected**  **healthy**

**are all mis-prediction costs equal?**
## Patient Status Prediction

Error measure = society cost

<table>
<thead>
<tr>
<th>actual</th>
<th>predicted</th>
<th>H7N9</th>
<th>cold</th>
<th>healthy</th>
</tr>
</thead>
<tbody>
<tr>
<td>H7N9</td>
<td></td>
<td>0</td>
<td>1000</td>
<td>100000</td>
</tr>
<tr>
<td>cold</td>
<td>100</td>
<td>0</td>
<td></td>
<td>3000</td>
</tr>
<tr>
<td>healthy</td>
<td>100</td>
<td>30</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

- H7N9 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 romance but likes terror**

<table>
<thead>
<tr>
<th>Error Measure = non-satisfaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual</td>
</tr>
<tr>
<td>romance</td>
</tr>
</tbody>
</table>

**customer 2 who likes terror and romance**

| actual            | predicted | romance | fiction | terror |
| romance          | romance   | 0       | 5       | 3      |

different customers:

**evaluate mis-predictions differently**
Cost-Sensitive Multiclass Classification Tasks

### Movie Classification with Non-Satisfaction

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>Romance</th>
<th>Fiction</th>
<th>Terror</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer 1, Romance</td>
<td>0</td>
<td>5</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>Customer 2, Romance</td>
<td>0</td>
<td>5</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

### Patient Diagnosis with Society Cost

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>H7N9</th>
<th>Cold</th>
<th>Healthy</th>
</tr>
</thead>
<tbody>
<tr>
<td>H7N9</td>
<td>0</td>
<td>1000</td>
<td>100000</td>
<td></td>
</tr>
<tr>
<td>Cold</td>
<td>100</td>
<td>0</td>
<td>30000</td>
<td></td>
</tr>
<tr>
<td>Healthy</td>
<td>100</td>
<td>30</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

### Check Digit Recognition with Absolute Cost

\[ C(y, h(x)) = |y - h(x)| \]
Cost Vector

cost vector $c$: a row of cost components

- customer 1 on a romance movie: $c = (0, 5, 100)$
- an H7N9 patient: $c = (0, 1000, 100000)$
- absolute cost for digit 2: $c = (1, 0, 1, 2)$
- “regular” classification cost for label 2: $c_c^{(2)} = (1, 0, 1, 1)$

regular classification: special case of cost-sensitive classification
Setup: Matrix-Based Cost-Sensitive Binary Classification

**Given**

\[ N \text{ examples, each } (\text{input } x_n, \text{label } y_n) \in \mathcal{X} \times \{1, 2, \ldots, K\} \]

and cost matrix \( C \in \mathbb{R}^{K \times K} \)

—will assume \( C(y, y) = 0 = \min_{1 \leq k \leq K} C(y, k) \)

**Goal**

a classifier \( g(x) \) that

pays a small cost \( C(y, g(x)) \)

on future **unseen** example \((x, y)\)

extension of ‘class-weighted’ cost-sensitive binary classification
Cost-Sensitive Multiclass Classification

Setup: Vector-Based Cost-Sensitive Binary Classification

Given

\[ N \text{ examples, each } (\text{input } x_n, \text{label } y_n) \in \mathcal{X} \times \{1, 2, \ldots, K\} \]

and cost vector \( c_n \in \mathbb{R}^K \)

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

extension of ‘example-weighted’ cost-sensitive binary classification
Which Age-Group?

- small mistake—classify a child as a teen;
  big mistake—classify an infant as an adult

- cost matrix $C(y, g(x))$ for embedding ‘order’: $C = \begin{pmatrix}
0 & 1 & 4 & 5 \\
1 & 0 & 1 & 3 \\
3 & 1 & 0 & 2 \\
5 & 4 & 1 & 0
\end{pmatrix}$

cost-sensitive classification can help solve many other problems, such as **ordinal ranking**
Outline

Cost-Sensitive Binary Classification

Bayesian Perspective of Cost-Sensitive Binary Classification

Non-Bayesian Perspective of Cost-Sensitive Binary Classification

Cost-Sensitive Multiclass Classification

Bayesian Perspective of Cost-Sensitive Multiclass Classification

Cost-Sensitive Classification by Reweighting and Relabeling

Cost-Sensitive Classification by Binary Classification

Cost-Sensitive Classification by Regression

Cost-and-Error-Sensitive Classification with Bioinformatics Application

Cost-Sensitive Ordinal Ranking with Information Retrieval Application

Summary
Key Idea: Conditional Probability Estimator

**Goal (Matrix Setup)**

A classifier \( g(\mathbf{x}) \) that pays a small cost \( C(y, g(\mathbf{x})) \) on future **unseen** example \((\mathbf{x}, y)\)

- **If** \( P(y|\mathbf{x}) \) known
  - Bayes optimal \( g^*(\mathbf{x}) = \arg\min_{1\leq k\leq K} \sum_{y=1}^{K} P(y|\mathbf{x})C(y, k) \)
- **If** \( p(y, \mathbf{x}) \approx P(y|\mathbf{x}) \) well
  - Approximately good \( g_p(\mathbf{x}) = \arg\min_{1\leq k\leq K} \sum_{y=1}^{K} p(y, \mathbf{x})C(y, k) \)

**How to get conditional probability estimator \( p \)?**

Logistic regression, Naïve Bayes, \ldots
Approximate Bayes-Optimal Decision

**if** $p(y, x) \approx P(+1|x)$ **well**

(Domingos, 1999)

approximately good $g_p(x) = \arg\min_{k \in \{1, 2, ..., K\}} \sum_{y=1}^{K} p(y, x)C(y, k)$

**Approximate Bayes-Optimal Decision (ABOD) Approach**

1. use your favorite algorithm on $\{(x_n, y_n)\}$ to get $p(y, x) \approx P(y|x)$
2. for each new input $x$, predict its class using $g_p(x)$ above

a simple extension from binary classification:
*probability estimate* + Bayes-optimal decision
ABOD on Artificial Data

1. use your favorite algorithm on \( \{(x_n, y_n)\} \) to get \( p(y, x) \approx P(y|x) \)
2. for each new input \( x \), predict its class using \( g_p(x) \)

LogReg

\[
\begin{array}{cccc}
1 & 2 & 3 & 4 \\
1 & 0 & 1 & 2 & 4 \\
2 & 4 & 0 & 1 & 2 \\
3 & 2 & 4 & 0 & 1 \\
4 & 1 & 2 & 4 & 0 \\
\end{array}
\]
Pros and Cons of ABOD

**Pros**
- optimal: if good probability estimate: \( p(y, x) \) really close to \( P(y|x) \)
- simple: with training (probability estimate) unchanged, and prediction (threshold) changed only a little

**Cons**
- ‘difficult’: good probability estimate often more difficult than good multiclass classification
- ‘restricted’: only applicable to class-weighted setup —need ‘full picture’ of cost matrix
- ‘slow prediction’: need sophisticated calculation at prediction stage

Can we use any multiclass classification algorithm for ABOD?
MetaCost Approach

Approximate Bayes-Optimal Decision (ABOD) Approach

1. use your favorite algorithm on \{ (x_n, y_n) \} to get \( p(y, x) \approx P(y|x) \)
2. for each new input \( x \), predict its class using \( g_p(x) \)

MetaCost Approach (Domingos, 1999)

1. use your favorite multiclass classification algorithm on bootstrapped \{ (x_n, y_n) \} and aggregate the classifiers to get \( p(y, x) \approx P(y|x) \)
2. for each given input \( x_n \), relabel it to \( y'_n \) using \( g_p(x) \)
3. run your favorite multiclass classification algorithm on relabeled \{ (x_n, y'_n) \} to get final classifier \( g \)
4. for each new input \( x \), predict its class using \( g(x) \)

pros: any multiclass classification algorithm can be used
MetaCost on Semi-Real Data

(Domingos, 1999)
- some “random” cost with UCI data
- MetaCost+C4.5: cost-sensitive
- C4.5: regular

not surprisingly,

considering the cost properly does help
Outline

Cost-Sensitive Binary Classification

Bayesian Perspective of Cost-Sensitive Binary Classification

Non-Bayesian Perspective of Cost-Sensitive Binary Classification

Cost-Sensitive Multiclass Classification

Bayesian Perspective of Cost-Sensitive Multiclass Classification

Cost-Sensitive Classification by Reweighting and Relabeling

Cost-Sensitive Classification by Binary Classification

Cost-Sensitive Classification by Regression

Cost-and-Error-Sensitive Classification with Bioinformatics Application

Cost-Sensitive Ordinal Ranking with Information Retrieval Application

Summary
Recall: Example-Weighting Useful for Binary

can example weighting be used for multiclass?

Yes! an elegant solution if using cost matrix with special properties (Zhou, 2010)

\[
\frac{C(i, j)}{C(j, i)} = \frac{w_i}{w_j}
\]

what if using cost vectors without special properties?
Key Idea: Cost Transformation

\[ \begin{pmatrix} 0 & 1000 \\ c \end{pmatrix} = \begin{pmatrix} 1000 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \]

\# of copies

classification costs

\[ \begin{pmatrix} 3 & 2 & 3 & 4 \\ \text{cost } c \end{pmatrix} = \begin{pmatrix} 1 & 2 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \]

mixture weights \( q_\ell \)

classification costs

- **split** the cost-sensitive example:
  \((x, 2)\) with \(c = (3, 2, 3, 4)\) equivalent to
  a weighted mixture \(\{(x, 1, 1), (x, 2, 2), (x, 3, 1)\}\)

**cost equivalence:**
\[ c[h(x)] = \sum_{\ell=1}^{K} q_\ell [\ell \neq h(x)] \text{ for any } h \]
Meaning of Cost Equivalence

\[ c[h(x)] = \sum_{\ell=1}^{K} q_\ell \mathbb{1}[\ell \neq h(x)] \]

- **on one** \((x, y, c)\)
  - wrong prediction charged by \(c[h(x)]\)

- **on all** \((x, \ell, q_\ell)\)
  - wrong prediction charged by total weighted classification error — weighted classification

Weighted classification \(\leadsto\) regular classification?

- same as binary (with CPEW) when \(q_\ell \geq 0\)

\[
\min_g \text{ expected LHS} = \min_g \text{ expected RHS}
\]

- (original cost-sensitive problem)
- (a regular problem when \(q_\ell \geq 0\))
Cost Transformation Methodology: Preliminary

1. split each training example \((x_n, y_n, c_n)\) to a weighted mixture \(\{(x_n, \ell, q_{n,\ell})\}_{\ell=1}^{K}\)
2. apply regular/weighted classification algorithm on the weighted mixtures \(\bigcup_{n=1}^{N} \{(x_n, \ell, q_{n,\ell})\}_{\ell=1}^{K}\)

- by \(c[g(x)] = \sum_{\ell=1}^{K} q_{\ell} [\ell \neq g(x)]\) (cost equivalence),
  good \(g\) for new regular classification problem
  \(\Rightarrow\) good \(g\) for original cost-sensitive classification problem
- regular classification: needs \(q_{n,\ell} \geq 0\)

but what if \(q_{n,\ell}\) negative?
Similar Cost Vectors

\[
\begin{pmatrix}
1 & 0 & 1 & 2 \\
3 & 2 & 3 & 4
\end{pmatrix}
= \begin{pmatrix}
1/3 & 4/3 & 1/3 & -2/3 \\
1 & 2 & 1 & 0
\end{pmatrix}
\cdot
\begin{pmatrix}
0 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 0
\end{pmatrix}
\]

- negative \( q_\ell \): cannot split
- but \( \hat{c} = (1, 0, 1, 2) \) is similar to \( c = (3, 2, 3, 4) \):
  for any classifier \( g \),

\[
\hat{c}[g(x)] + \text{constant} = c[g(x)] = \sum_{\ell=1}^{K} q_\ell \mathbb{1}[\ell \neq g(x)]
\]

- constant can be dropped during minimization

\[
\min_g \text{ expected } \hat{c}[g(x)] \quad \text{(original cost-sensitive problem)}
= \min_g \text{ expected LHS} \quad \text{(a regular problem when } q_\ell \geq 0)\]
Cost Transformation Methodology: Revised

1. shift each training cost $\hat{c}_n$ to a similar and “splittable” $c_n$

2. split $(x_n, y_n, c_n)$ to a weighted mixture $\{ (x_n, \ell, q_n, \ell) \}_{\ell=1}^K$

3. apply regular classification algorithm on the weighted mixtures

$$\bigcup_{n=1}^{N} \{ (x_n, \ell, q_n, \ell) \}_{\ell=1}^K$$

- **splittable**: $q_{n,\ell} \geq 0$

- by cost equivalence after shifting:

  good $g$ for new regular classification problem

  $\equiv$ good $g$ for original cost-sensitive classification problem

but infinitely many similar and splittable $c_n$!
Uncertainty in Mixture

- a single example $\{(x, 2)\}$ —**certain** that the desired label is 2
- a mixture $\{(x, 1, 1), (x, 2, 2), (x, 3, 1)\}$ sharing the same $x$ —**uncertainty** in the desired label (25%: 1, 50%: 2, 25%: 3)
- over-shifting adds unnecessary mixture uncertainty:

\[
\begin{pmatrix}
3 & 2 & 3 & 4 \\
33 & 32 & 33 & 34
\end{pmatrix}
\quad =
\begin{pmatrix}
1 & 2 & 1 & 0 \\
11 & 12 & 11 & 10
\end{pmatrix}
\begin{pmatrix}
0 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 0
\end{pmatrix}
\]

should choose a similar and splittable $c$ with **minimum mixture uncertainty**
Cost Transformation Methodology (Lin, 2010)

1. Shift each training cost \( \hat{c}_n \) to a similar and splittable \( c_n \) with minimum "mixture uncertainty"
2. Split \( (x_n, y_n, c_n) \) to a weighted mixture \( \{(x_n, \ell, q_n, \ell)\}_{\ell=1}^{K} \)
3. Apply regular classification algorithm on the weighted mixtures

\[
\bigcup_{n=1}^{N} \left\{ (x_n, \ell, q_n, \ell) \right\}_{\ell=1}^{K}
\]

- Mixture uncertainty: entropy of normalized \( (q_1, q_2, \ldots, q_K) \)
- A simple and unique optimal shifting exists for every \( \hat{c} \)

Good \( g \) for new regular classification problem
\( \equiv \) Good \( g \) for original cost-sensitive classification problem
Data Space Expansion Approach (DSE) Approach (Abe, 2004)

1. For each \((x_n, y_n, c_n)\) and \(\ell\), let
   \[ q_{n,\ell} = \max_{1 \leq k \leq K} c_n[k] - c_n[\ell] \]

2. Apply your favorite multiclass classification algorithm on the weighted mixtures
   \[ \bigcup_{n=1}^{N} \{(x_n, \ell, q_{n,\ell})\} \]
   to get \(g(x)\)

3. For each new input \(x\), predict its class using \(g(x)\)

- Detailed explanation provided by the cost transformation methodology discussed above (Lin, 2010)
- Extension of Cost-Proportionate Example Weighting, but now with relabeling!

Pros: any multiclass classification algorithm can be used
### DSE versus MetaCost on Semi-Real Data

(Abe, 2004)

<table>
<thead>
<tr>
<th></th>
<th>MetaCost</th>
<th>DSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>annealing</td>
<td>206.8</td>
<td>127.1</td>
</tr>
<tr>
<td>solar</td>
<td>5317</td>
<td>110.9</td>
</tr>
<tr>
<td>kdd99</td>
<td>49.39</td>
<td>46.68</td>
</tr>
<tr>
<td>letter</td>
<td>129.6</td>
<td>114.0</td>
</tr>
<tr>
<td>splice</td>
<td>49.95</td>
<td>135.5</td>
</tr>
<tr>
<td>satellite</td>
<td>104.4</td>
<td>116.8</td>
</tr>
</tbody>
</table>

- some “random” cost with UCI data
- C4.5 with COSTING for weighted classification

DSE comparable to MetaCost
Cons of DSE: Unavoidable (Minimum) Uncertainty

Original Cost-Sensitive Classification Problem

- individual examples with certainty

New Regular Classification Problem

- mixtures with unavoidable uncertainty

- cost embedded as weight + label
- new problem usually **harder** than original one

need **robust** multiclass classification algorithm to deal with uncertainty
Outline

Cost-Sensitive Binary Classification
Bayesian Perspective of Cost-Sensitive Binary Classification
Non-Bayesian Perspective of Cost-Sensitive Binary Classification
Cost-Sensitive Multiclass Classification
Bayesian Perspective of Cost-Sensitive Multiclass Classification
Cost-Sensitive Classification by Reweighting and Relabeling

Cost-Sensitive Classification by Binary Classification
Cost-Sensitive Classification by Regression
Cost-and-Error-Sensitive Classification with Bioinformatics Application
Cost-Sensitive Ordinal Ranking with Information Retrieval Application
Summary
Key Idea: Design Robust Multiclass Algorithm

One-Versus-One: A Popular Classification Meta-Method

1. for a pair \((i, j)\), take all examples \((x_n, y_n)\) that \(y_n = i\) or \(j\) (original one-versus-one)

2. for a pair \((i, j)\), from each weighted mixture \(\{(x_n, \ell, q_n,\ell)\}\) with \(q_{n,i} > q_{n,j}\), take \((x_n, i)\) with weight \(q_{n,i} - q_{n,j}\); vice versa (robust one-versus-one)

3. train a binary classifier \(\hat{g}^{(i,j)}\) using those examples

4. repeat the previous two steps for all different \((i, j)\)

5. predict using the votes from \(\hat{g}^{(i,j)}\)

- un-shifting inside the meta-method to remove uncertainty
- robust step makes it suitable for cost transformation methodology

**cost-sensitive one-versus-one:**
cost transformation + robust one-versus-one
for a pair \((i, j)\), transform all examples \((x_n, y_n)\) to
\[
\left( x_n, \arg\min_{k \in \{i, j\}} c_n[k] \right)
\]
with weight \(|c_n[i] - c_n[j]|\)

2. train a binary classifier \(\hat{g}^{(i,j)}\) using those examples

3. repeat the previous two steps for all different \((i, j)\)

4. predict using the votes from \(\hat{g}^{(i,j)}\)

- comes with **good theoretical guarantee**:
  \[
  \text{test cost of final classifier} \leq 2 \sum_{i < j} \text{test cost of } \hat{g}^{(i,j)}
  \]

- **simple, efficient**, and takes original OVO as **special case**

**physical meaning:**

each \(\hat{g}^{(i,j)}\) answers yes/no question “prefer \(i\) or \(j\)?”
CSOVO on Semi-Real Data

(Lin, 2010)

- some “random” cost with UCI data
- CSOVO-SVM: cost-sensitive
- OVO-SVM: regular

not surprisingly again,

considering the cost properly does help
CSOVO for Ordinal Ranking

(Lin, 2010)

- absolute cost with benchmark ordinal ranking data
- **CSOVO-SVM**: cost-sensitive
- **OVO-SVM**: regular

**CSOVO significantly better for ordinal ranking**
Other Approaches via Weighted Binary Classification

Filter Tree (FT): $K - 1$ binary classifiers (Beygelzimer, 2007)

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

Weighted All Pairs (WAP): $\frac{K(K-1)}{2}$ binary classifiers (Beygelzimer, 2005)

- Similar to CSOVO, with theoretically better way of calculating weights

Sensitive Error Correcting Output Code (SECOC): $(T \cdot K)$ binary classifiers (Langford, 2005)


Extended Binary Classification: $K$ binary classifiers (Lin, 2012)

- Is lowest-cost $y \leq$ some $k$?
- More proper for ordinal ranking
Outline

Cost-Sensitive Binary Classification
Bayesian Perspective of Cost-Sensitive Binary Classification
Non-Bayesian Perspective of Cost-Sensitive Binary Classification
Cost-Sensitive Multiclass Classification
Bayesian Perspective of Cost-Sensitive Multiclass Classification
Cost-Sensitive Classification by Reweighting and Relabeling
Cost-Sensitive Classification by Binary Classification
Cost-Sensitive Classification by Regression
Cost-and-Error-Sensitive Classification with Bioinformatics Application
Cost-Sensitive Ordinal Ranking with Information Retrieval Application
Summary
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**

Optimal

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

**if $r_k(x) \approx c[k]$ well**

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

<table>
<thead>
<tr>
<th>input (x_1)</th>
<th>(c_n[1])</th>
<th>input (x_1)</th>
<th>(c_n[2])</th>
<th>(r_1)</th>
<th>input (x_1)</th>
<th>(c_n[K])</th>
<th>(r_K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_1)</td>
<td>0,</td>
<td>(x_1)</td>
<td>2,</td>
<td>(r_1)</td>
<td>(x_1)</td>
<td>1</td>
<td>(r_K)</td>
</tr>
<tr>
<td>(x_2)</td>
<td>1,</td>
<td>(x_2)</td>
<td>3,</td>
<td>(r_2)</td>
<td>(x_2)</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>(x_N)</td>
<td>6,</td>
<td>(x_N)</td>
<td>1,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

want: \(r_k(x) \approx c[k]\) for all future \((x, y, c)\) and \(k\)
Cost-Sensitive Classification by Regression

The Reduction Framework

1. Transform cost-sensitive examples \((x_n, y_n, c_n)\) to regression examples \((x_{n,k}, Y_{n,k}) = (x_n, c_n[k])\).

2. Use your favorite algorithm on the regression examples and get estimators \(r_k(x)\).

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

The reduction-to-regression framework: systematic & easy to implement.
Theoretical Guarantees (1/2)

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

**Theorem (Absolute Loss Bound)**

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

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

**low-cost classifier \(\iff\) accurate estimator**
Theoretical Guarantees (2/2)

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

**Theorem (Squared Loss Bound)**

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

\[
    c[g_r(x)] \leq \sqrt{2 \sum_{k=1}^{K} (r_k(x) - c[k])^2}.
\]

applies to common least-square regression
Cost-Sensitive Classification by Regression

A Pictorial Proof

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

- Assume \( \mathbf{c} \) ordered and not degenerate:
  \( y = 1; 0 = \mathbf{c}[1] < \mathbf{c}[2] \leq \cdots \leq \mathbf{c}[K] \)

- Assume mis-prediction \( g_r(\mathbf{x}) = 2 \):
  \( r_2(\mathbf{x}) = \min_{1 \leq k \leq K} r_k(\mathbf{x}) \leq r_1(\mathbf{x}) \)

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

1. $\Delta_1 \geq 0$ and $\Delta_2 \geq 0$: $c[2] \leq \Delta_1 + \Delta_2$
2. $\Delta_1 \leq 0$ and $\Delta_2 \geq 0$: $c[2] \leq \Delta_2$
3. $\Delta_1 \geq 0$ and $\Delta_2 \leq 0$: $c[2] \leq \Delta_1$

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

![Diagram](image-url)
Cost-Sensitive Classification by Regression

Tighter Bound with One-sided Loss

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

with

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

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

**Intuition**

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

—both wishes same as \( \Delta_k \leq 0 \) and hence \( \xi_k = 0 \)

**One-sided Loss Bound:**

\[
c[g_r(x)] \leq \sum_{k=1}^{K} \xi_k \leq \sum_{k=1}^{K} |\Delta_k|
\]
Cost-Sensitive Classification by Regression

The Improved Reduction Framework

1. transform cost-sensitive examples \((x_n, y_n, c_n)\) to regression examples
2. use a one-sided regression algorithm to get estimators \(r_k(x)\)
3. for each new input \(x\), predict its class using \(g_r(x) = \text{argmin}_{1 \leq k \leq K} r_k(x)\)

(Tu, 2010)

the reduction-to-OSR framework: need a good OSR algorithm
Regularized One-sided Hyper-linear Regression

Given

\[(x_{n,k}, Y_{n,k}, Z_{n,k}) = (x_n, c_n[k], 2[c_n[k] = c_n[y_n]] - 1)\]

Training Goal

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

—will drop \(k\)

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

to get \(r_k(x) = \langle w, \phi(x) \rangle + b\)
One-sided Support Vector Regression

Regularized One-sided Hyper-linear Regression

\[
\min_{\mathbf{w}, b} \quad \frac{\lambda}{2} \langle \mathbf{w}, \mathbf{w} \rangle + \sum_{n=1}^{N} \xi_n \\
\xi_n = \max \left( \mathbf{Z}_n \cdot (r_k(\mathbf{x}_n) - Y_n), 0 \right)
\]

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 + \(0 \rightarrow \epsilon\) + (keep \(\xi_n\) or \(\xi^*_n\) by \(\mathbf{Z}_n\))
OSR-SVM: $g_r(x) = \text{argmin } r_k(x)$

$$
\min_{w,b} \frac{\lambda}{2} \langle w, w \rangle + \sum_{n=1}^{N} \xi_n \\
\text{with } r_k(x) = \langle w, \phi(x) \rangle + b \\
\xi_n = \max (Z_n \cdot (r_k(x_n) - Y_n), 0)
$$

OVA-SVM: $g_r(x) = \text{argmax } q_k(x)$

$$
\text{with } q_k(x) = \langle w, \phi(x) \rangle + b \\
\xi_n = \max (-Z_n \cdot q_k(x_n) + 1, 0)
$$

OVA-SVM: special case that replaces $Y_n$ (i.e. $c_n[k]$) by $-Z_n$
OSR-SVM on Semi-Real Data

(Tu, 2010)

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

OSR often significantly better than OVA
OSR versus FT on Semi-Real Data

(Tu, 2010)

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

FT faster, but OSR better performing
OSR versus WAP on Semi-Real Data

(Tu, 2010)

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

OSR faster and comparable performance
OSR versus SECOC on Semi-Real Data

(Tu, 2010)

- **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
Biased Personal Favorites

- OSR: fast training, fast prediction, very good performance
- WAP or CSOVO: stable performance, pretty strong theoretical guarantee
- FT: fast training, very fast prediction, good performance, strong theoretical guarantee
- MetaCost if in the mood for Bayesian :-)

Hsuan-Tien Lin (NTU CSIE)
Outline

Cost-Sensitive Binary Classification
Bayesian Perspective of Cost-Sensitive Binary Classification
Non-Bayesian Perspective of Cost-Sensitive Binary Classification
Cost-Sensitive Multiclass Classification
Bayesian Perspective of Cost-Sensitive Multiclass Classification
Cost-Sensitive Classification by Reweighting and Relabeling
Cost-Sensitive Classification by Binary Classification
Cost-Sensitive Classification by Regression
Cost-and-Error-Sensitive Classification with Bioinformatics Application
Cost-Sensitive Ordinal Ranking with Information Retrieval Application
Summary
A Real Medical Application: Classifying Bacteria

The Problem

- by human doctors: **different treatments** ↔ **serious costs**
- cost matrix averaged from two doctors:

```
<table>
<thead>
<tr>
<th></th>
<th>Ab</th>
<th>Ecoli</th>
<th>HI</th>
<th>KP</th>
<th>LM</th>
<th>Nm</th>
<th>Psa</th>
<th>Spn</th>
<th>Sa</th>
<th>GBS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ab</td>
<td>0</td>
<td>1</td>
<td>10</td>
<td>7</td>
<td>9</td>
<td>9</td>
<td>5</td>
<td>8</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>Ecoli</td>
<td>3</td>
<td>0</td>
<td>10</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>10</td>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>HI</td>
<td>10</td>
<td>10</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>10</td>
<td>1</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>KP</td>
<td>7</td>
<td>7</td>
<td>3</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>LM</td>
<td>8</td>
<td>8</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td>2</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>Nm</td>
<td>3</td>
<td>10</td>
<td>9</td>
<td>8</td>
<td>6</td>
<td>0</td>
<td>8</td>
<td>3</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Psa</td>
<td>7</td>
<td>8</td>
<td>10</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>7</td>
<td>0</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>Spn</td>
<td>6</td>
<td>10</td>
<td>7</td>
<td>7</td>
<td>4</td>
<td>4</td>
<td>9</td>
<td>0</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>Sa</td>
<td>7</td>
<td>10</td>
<td>6</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>9</td>
<td>2</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>GBS</td>
<td>2</td>
<td>5</td>
<td>10</td>
<td>9</td>
<td>8</td>
<td>6</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>0</td>
</tr>
</tbody>
</table>
```

is cost-sensitive classification **realistic**?
OSR best: cost-sensitive classification is helpful
Soft Cost-sensitive Classification

The Problem

• cost-sensitive classifier: low cost but high error
• traditional classifier: low error but high cost
• how can we get the blue classifiers?: low error and low cost

Cost-and-error-sensitive: more suitable for medical needs
**Improved OSR for Cost and Error on Semi-Real Data**

**key idea (Jan, 2012):** consider a ‘modified’ cost that mixes original cost and ‘regular cost’

**Cost**

<table>
<thead>
<tr>
<th></th>
<th>iris</th>
<th>wine</th>
<th>glass</th>
<th>vehicle</th>
<th>vowel</th>
<th>segment</th>
<th>dna</th>
<th>satimage</th>
<th>usps</th>
<th>zoo</th>
<th>splice</th>
<th>ecoli</th>
<th>soybean</th>
</tr>
</thead>
<tbody>
<tr>
<td>≈</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>≈</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>≈</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>≈</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>≈</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>≈</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>≈</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>≈</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Error**

<table>
<thead>
<tr>
<th></th>
<th>iris</th>
<th>wine</th>
<th>glass</th>
<th>vehicle</th>
<th>vowel</th>
<th>segment</th>
<th>dna</th>
<th>satimage</th>
<th>usps</th>
<th>zoo</th>
<th>splice</th>
<th>ecoli</th>
<th>soybean</th>
</tr>
</thead>
<tbody>
<tr>
<td>◯</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>◯</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>◯</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>◯</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>◯</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>◯</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>◯</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>◯</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>◯</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

improves other cost-sensitive classification algorithms, too

Hsuan-Tien Lin (NTU CSIE)
Outline

Cost-Sensitive Binary Classification
Bayesian Perspective of Cost-Sensitive Binary Classification
Non-Bayesian Perspective of Cost-Sensitive Binary Classification
Cost-Sensitive Multiclass Classification
Bayesian Perspective of Cost-Sensitive Multiclass Classification
Cost-Sensitive Classification by Reweighting and Relabeling
Cost-Sensitive Classification by Binary Classification
Cost-Sensitive Classification by Regression
Cost-and-Error-Sensitive Classification with Bioinformatics Application

Cost-Sensitive Ordinal Ranking with Information Retrieval Application

Summary
not just for searching good machine learning book :-);
but also for recommendation systems & other web service
Three Properties of Search-Engine Ranking

- listwise with focus on top ranks
  - query-oriented & personalized
  - emphasis on highly-preferred (relevant) items
- large scale
  - both during training & testing
  - e.g. Yahoo! Learning-To-Rank Challenge 2010: 473K training URLs, 166K test URLs
- ordinal data
  - labeled qualitatively by human, e.g. \{highly irrelevant, irrelevant, neutral, relevant, highly relevant\}
  - lack of quantitative info

search-engine ranking problem:
  learning a ranker from large scale ordinal data
  with focus on top ranks
Search-Engine Ranking Setup

Given
for query indices $q = 1, 2, \ldots, Q$,

- a set of related documents $\{x_{q,i}\}_{i=1}^{N(q)}$
- ordinal relevance $y_{q,i} \in \mathcal{Y} = \{0, 1, \ldots, K\}$ for each document $x_{q,i}$

with large $Q$ and $N(q)$

Goal
a ranker $r(x)$ that “accurately ranks” top $x_{Q+1,i}$ from an unseen set of documents $\{x_{Q+1,i}\}$

how to evaluate accurate ranking around the top?
Expected Reciprocal Rank \((\text{ERR}; \text{Chapelle, 2009})\)

Assume for any example \((\text{document } x, \text{rank } y)\),

\[
P(\text{user chooses document } x) = \frac{2^y - 1}{2^K}
\]

Assumption: Stopping Probability of List of Documents

\[
P(\text{user stops at position } i \text{ of list}) = P(\text{doesn't stop at pos. } i - 1) \times \frac{1}{i} P(\text{chooses document at pos. } i)
\]

ERR: Total Discounted Stopping Probability of List

\[
\text{ERR}_q(r) \equiv \sum_{i=1}^{N(q)} \frac{1}{i} P(\text{user stops at position } i \text{ of the list ordered by } r)
\]

large ERR ⇔ small \(i\) matches large \(P\)
⇔ good ranking around top
Cost-Sensitive Ordinal Classification via Regression (COCR)

- Reduction from listwise ranking (ERR) to cost-sensitive (ordinal) classification (approximately)
  — Aim for top rank and large scale data
- Reduction from cost-sensitive ordinal classification to binary classification
  — Aim for respecting ordinal data
- Reduction from binary classification to regression
  — Aim for large scale data and avoiding discrete ties

Costs can approximately embed true criteria of interest
Optimistic ERR (oERR) Cost for COCR

desired listwise criteria

How to make \( \text{ERR}(r) \) close to \( \text{ERR}(p) \), the ERR of perfect ranker?

embed criteria within cost

\[
\text{ERR}(p) - \text{ERR}(r) \leq \sum_{i=1}^{N(q)} \left( 2^{y_{q,i}} - 2^{r_{x_{q,i}}} \right)^2 + \Delta
\]

- \( \Delta \approx 0 \) if \( r \approx p \) (optimistic)
- then, \( c[k] = (2^y - 2^k)^2 \) embeds ERR
- oERR cost can then be coupled with other ordinal ranking techniques to improve performance

not a very tight bound, but better than nothing
### COCR on Benchmark Data

(Ruan, 2013)

<table>
<thead>
<tr>
<th>data set</th>
<th>Direct Regression</th>
<th>benchmark</th>
<th>oERR-COCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTRC1</td>
<td>0.4470</td>
<td>0.4484</td>
<td>0.4505</td>
</tr>
<tr>
<td>LTRC2</td>
<td>0.4440</td>
<td>0.4465</td>
<td>0.4461</td>
</tr>
<tr>
<td>MS10K</td>
<td>0.2643</td>
<td>0.2642</td>
<td>0.2792</td>
</tr>
<tr>
<td>MS30K</td>
<td>0.2748</td>
<td>0.2748</td>
<td>0.2942</td>
</tr>
</tbody>
</table>

- best ERR
- significantly better than direct regression

- oERR-COCR **usually the best**
Cost-Sensitive Binary Classification
Bayesian Perspective of Cost-Sensitive Binary Classification
Non-Bayesian Perspective of Cost-Sensitive Binary Classification
Cost-Sensitive Multiclass Classification
Bayesian Perspective of Cost-Sensitive Multiclass Classification
Cost-Sensitive Classification by Reweighting and Relabeling
Cost-Sensitive Classification by Binary Classification
Cost-Sensitive Classification by Regression
Cost-and-Error-Sensitive Classification with Bioinformatics Application
Cost-Sensitive Ordinal Ranking with Information Retrieval Application

Summary
Summary

- **cost-sensitive binary classification**: just the weights
  - Bayesian: Approximate Bayes Optimal Decision (Elkan, 2001)
  - non-Bayesian: Cost-Proportionate Example Weighting (Zadrozny, 2003)

- **cost-sensitive binary classification**: cost matrix/vectors
  - Bayesian: MetaCost (Domingos, 1999)
  - non-Bayesian:
    Data Space Expansion (Abe, 2004) (to multiclass),
    Cost-Sensitive One-Versus-One (Lin, 2012), ... (to binary),
    One-Sided Regression (Tu, 2010) (to regression)
  —most implemented here:
    http://www.csie.ntu.edu.tw/~htlin/program/cssvm/

- **beyond**:
  - cost-and-error-sensitive for medical application (Jan, 2012)
  - cost-sensitive, approximately, for information retrieval (Ruan, 2013)
  - cost-intervals (Liu, 2010)

Discussion welcomed on algorithm and **application** opportunities
Summary

Giants’ Shoulder

- **binary:**
  - Zadrozny et al., Cost-Sensitive Learning by Cost-Proportionate Example Weighting, 2003
  - Abu-Mostafa et al., Learning from Data: A Short Course, 2013

- **multiclass:**
  - Abe et al., An Iterative Method for Multi-Class Cost-Sensitive Learning, 2004
  - Beygelzimer et al., Error Limiting Reductions Between Classification Tasks, 2005
  - Langford and Beygelzimer, Sensitive Error Correcting Output Codes, 2005
  - Beygelzimer et al., Multiclass Classification with Filter Trees, 2007
  - Chapelle et al., Expected Reciprocal Rank for Graded Relevance, 2009
  - Tu and Lin, One-Sided Support Vector Regression for Multiclass Cost-Sensitive Classification, 2010
  - Lin, A Simple Cost-Sensitive Multiclass Classification Algorithm Using One-Versus-One Comparisons, 2010
  - Jan et al., Cost-Sensitive Classification on Pathogen Species of Bacterial Meningitis by Surface Enhanced Raman Scattering, 2011
  - Lin and Li, Reduction from Cost-Sensitive Ordinal Ranking to Weighted Binary Classification, 2012
  - Jan et al., A Simple Methodology for Soft Cost-Sensitive Classification, 2012
  - Ruan et al., Improving Ranking Performance with Cost-Sensitive Ordinal Classification via Regression, 2013
Acknowledgments

- ACML Organizers!
- Computational Learning Lab @ NTU and Learning Systems Group @ Caltech for discussions

final advertisement 😊:
my student’s work on bipartite ranking (last talk of the conference)

Wei-Yuan Shen and Hsuan-Tien Lin. Active Sampling of Pairs and Points for Large-scale Linear Bipartite Ranking. ACML 2013.

Thank you. Questions?