# Machine Learning Reductions Tutorial

John Langford (with help from many others...)

Yahoo Research

Background: http://hunch.net/~jl/projects/reductions/reductions.html

Helsinki 2006, Finland, Oct 12

Idea: Reuse old learning algos to solve new problems

Evidence from the neuroside.

1. Rewired Ferrets

2. Brain Damage in the Young

Basic question: What can be solved in a reductionist way?

## Characteristics of Learning Reductions

1. Reductionist.
2. Elemental.
3. Works well in practice.
4. Easy.

## It's reductionist (= good research direction)

Reductionist = cut problems into small problems, solve small problems, and compose to solve big problem.

Some other reductionist things:

1. The transistor for computations

2. Rendering triangles for rendering scenes

3. Much of science

## Elemental

Reduction needs to know nothing about oracle learning algorithm except its type $\Rightarrow$Modularity, code reuse, universality.

1. Can reuse old learning algorithms

2. Can reuse old code

It's easy (= you can use it too)

The reductions method to solving learning problems:

1. Identify the type of learning problem $B$.
2. Find premade reduction $R$ and oracle learning algorithm $A$.
3. Build a $B$ predictor using $R^A$ + data.

Given a Binary classifier, how can we solve

1. Importance Weighted Classification?

2. Regression?

3. Multiclass Classification?

4. Cost Sensitive Classification?

## Classification Definition

- Problem: A measure $D$ on $X \times \{0,1\}$ where $X$ is an arbitrary space.
- Classifier: $c : X \to \{0,1\}$ = predictor
- Given $S = (X \times \{0,1\})^*$ find classifier $c$ with small error rate

$$e(D,c) = \Pr_{(x,y)\sim D}(c(x) \neq y)$$

Note: $D$ unknown. Impossible in general, but maybe possible in particular. We hope $S$ drawn IID from $D$, but it isn't always so.

## Importance Weighted Classification

- Problem: A measure $D$ on $X \times \{0,1\} \times [0,\infty)$ where $X$ is an arbitrary space.

- Classifier: $c : X \to \{0,1\}$ = predictor

- Given $S = (X \times \{0,1\} \times [0,\infty))^*$ find classifier $c$ with small importance weighted loss

$$e_w(D,c) = E_{(x,y,i)\sim D}[iI(c(x) \neq y)]$$

## The core theorem: folklore

Theorem: (Distribution shift) For all $c$, for all importance weighted $D$, let $D'(x,y,i) = \frac{iD(x,y,i)}{E_{(x,y,i)\sim D}[i]}$. Then:

$$e_w(D,c) = e(D',c)E_{(x,y,i)\sim D}[i]$$

... so minimizing $D$ error rate = minimizing $D_i$ importance weighted error rate. Proof:

$e_w(D,c) = \sum_{(x,y,i)}[iD(x,y,i)I(c(x) \neq y)]$

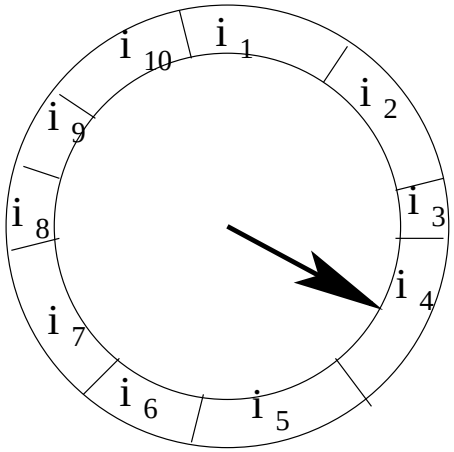
$= E_{(x,y,i)\sim D}[i] \sum_{(x,y,i)}[D'(x,y,i)I(c(x) \neq y)]$

$= E_{(x,y,i)\sim D}[i] \Pr_{(x,y)\sim D'}[I(c(x) \neq y)]$

$= E_{(x,y,i)\sim D}[i]e(D',c)$

How do we change distributions?

Something which doesn't work: Resampling




Resampling=place examples on roulette wheel with coverage proportional to weight. Spin the wheel many times.

Basic problem: duplicate examples = nonindependence.

## Distribution Transform: Rejection Sampling.

1. Pick a constant $c$ larger than any importance ($\forall i \;\; c > i$)

2. For each sample $(x, y, i)$, flip a coin with bias $\frac{i}{c}$. If the result is "heads" keep it, and otherwise discard it.

Rejection sampling $\Rightarrow$ samples in $S$ are IID if samples in $S_w$ are IID.

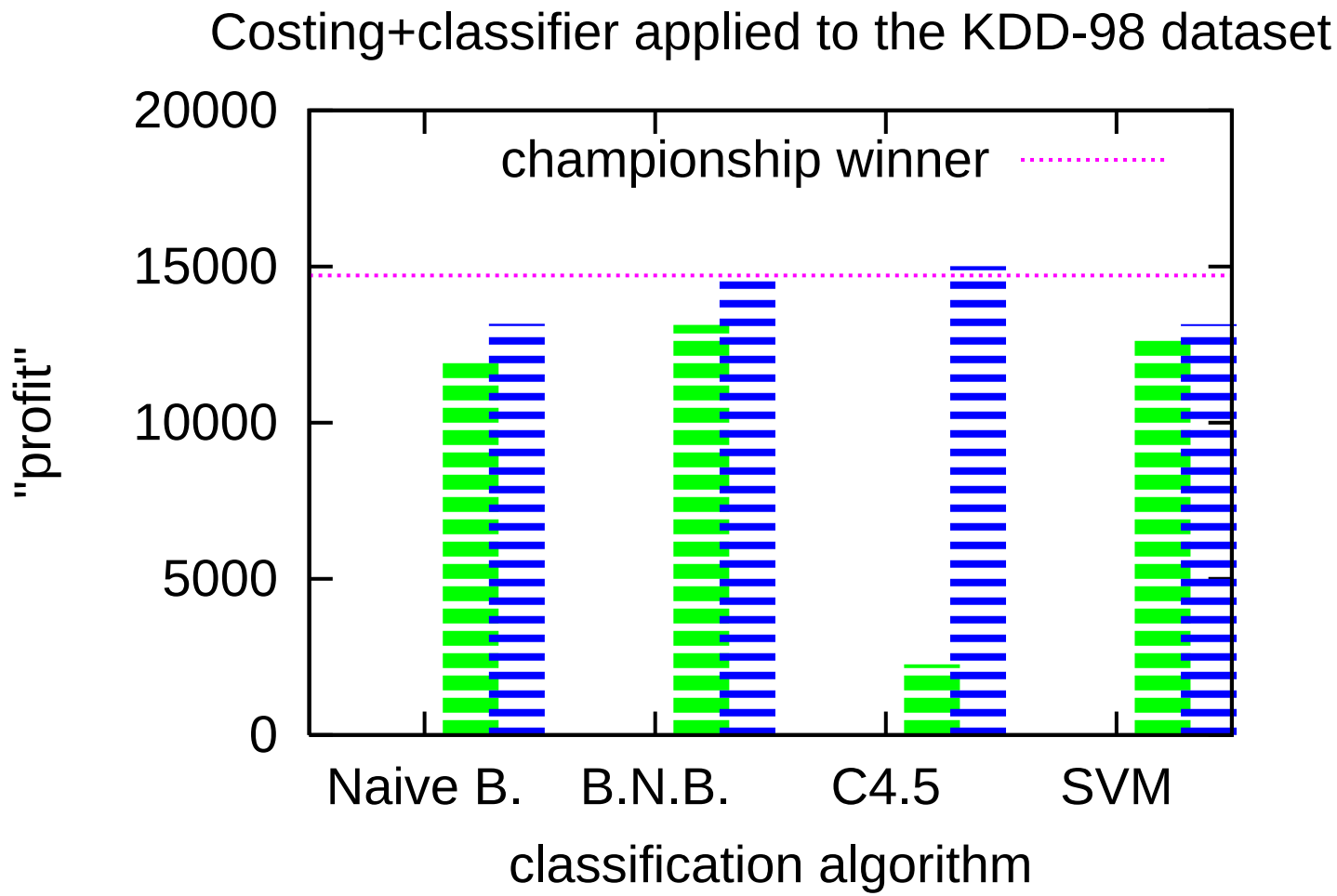
Costing($S_w$,$A$)

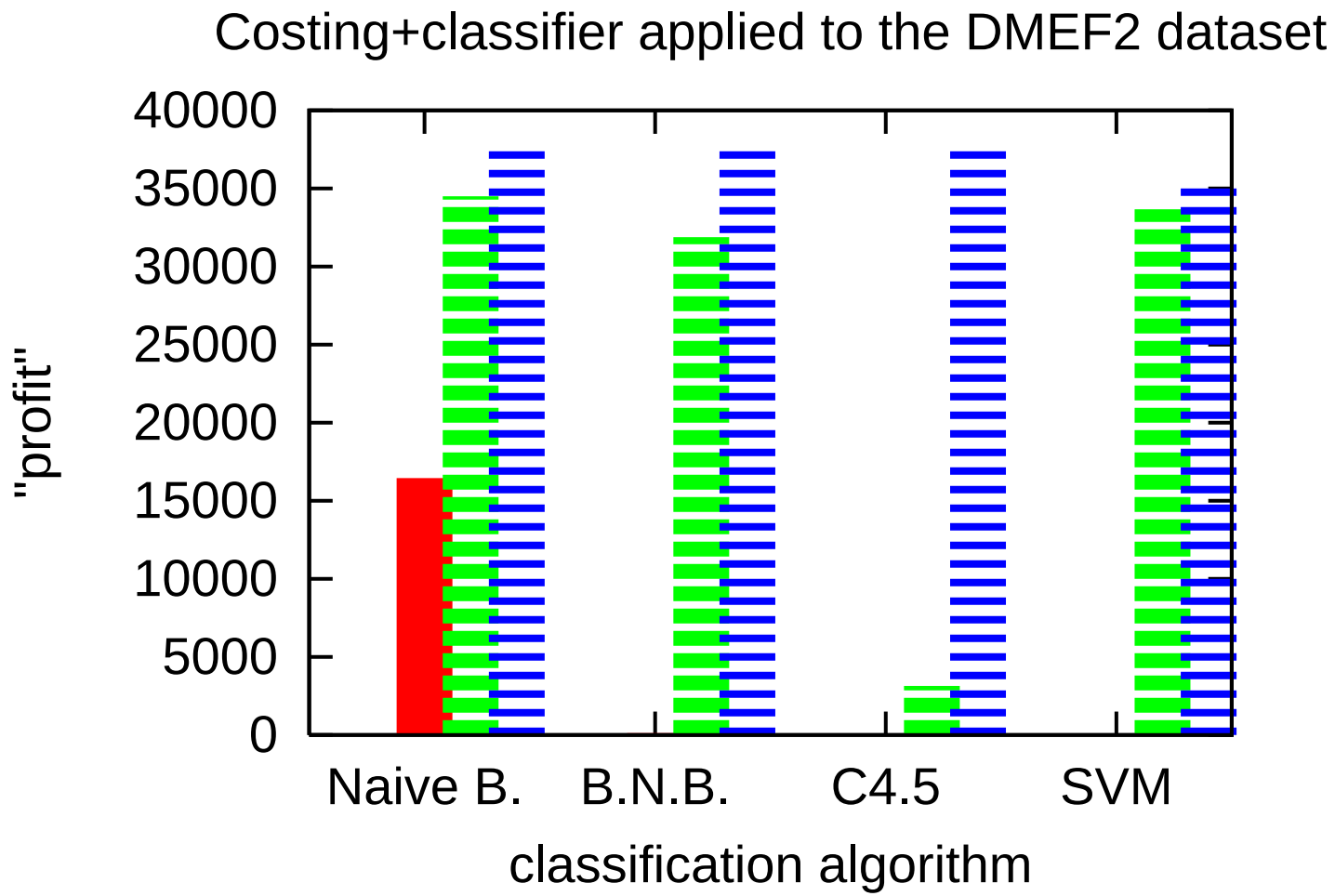
1. For $t = 1$ to $10$ do

   (a) Rejection sample to form $S_t$ from $S_w$.

   (b) Learn $c_t = A(S_t)$

Output: $c(x) = \text{majority}(\{c_1(x), ..., c_{10}(x)\})$


Costing+classifier applied to the KDD-98 dataset
20000
15000
10000
5000
0
"profit"
championship winner
Naive B.
B.N.B.
C4.5
SVM
classification algorithm


Costing+classifier applied to the DMEF2 dataset
"profit"
40000
35000
30000
25000
20000
15000
10000
5000
0
Naive B.
B.N.B.
C4.5
SVM
classification algorithm

Given a Binary classifier, how can we solve

1. Importance Weighted Classification?

2. Regression?

3. Multiclass Classification?

4. Cost Sensitive Classification?

## Square Error Regression

- Problem: A measure $D$ on $X \times [0,1]$ where $X$ is an arbitrary space.

- A Regressor: $h : X \to [0,1]$ = predictor

- Given $S = (X \times [0,1])^*$ find regressor $h$ with small squared error:

$$e_r(D,h) = E_{(x,y)\sim D}[(h(x)-y)^2]$$

## Reasons for the Regression Problem

1. Doctor wants "advice" from a machine, but not a decision.

2. Distributed system requires efficient communication of beliefs.

3. Compatibility between prediction and probabilistic prediction worlds.

# The Probing Method: Observations

1. Pick $t \in [0, 1]$.

2. Map $(x, y) \rightarrow (x, I(y > t), |y - t|)$ (= importance weighted example)

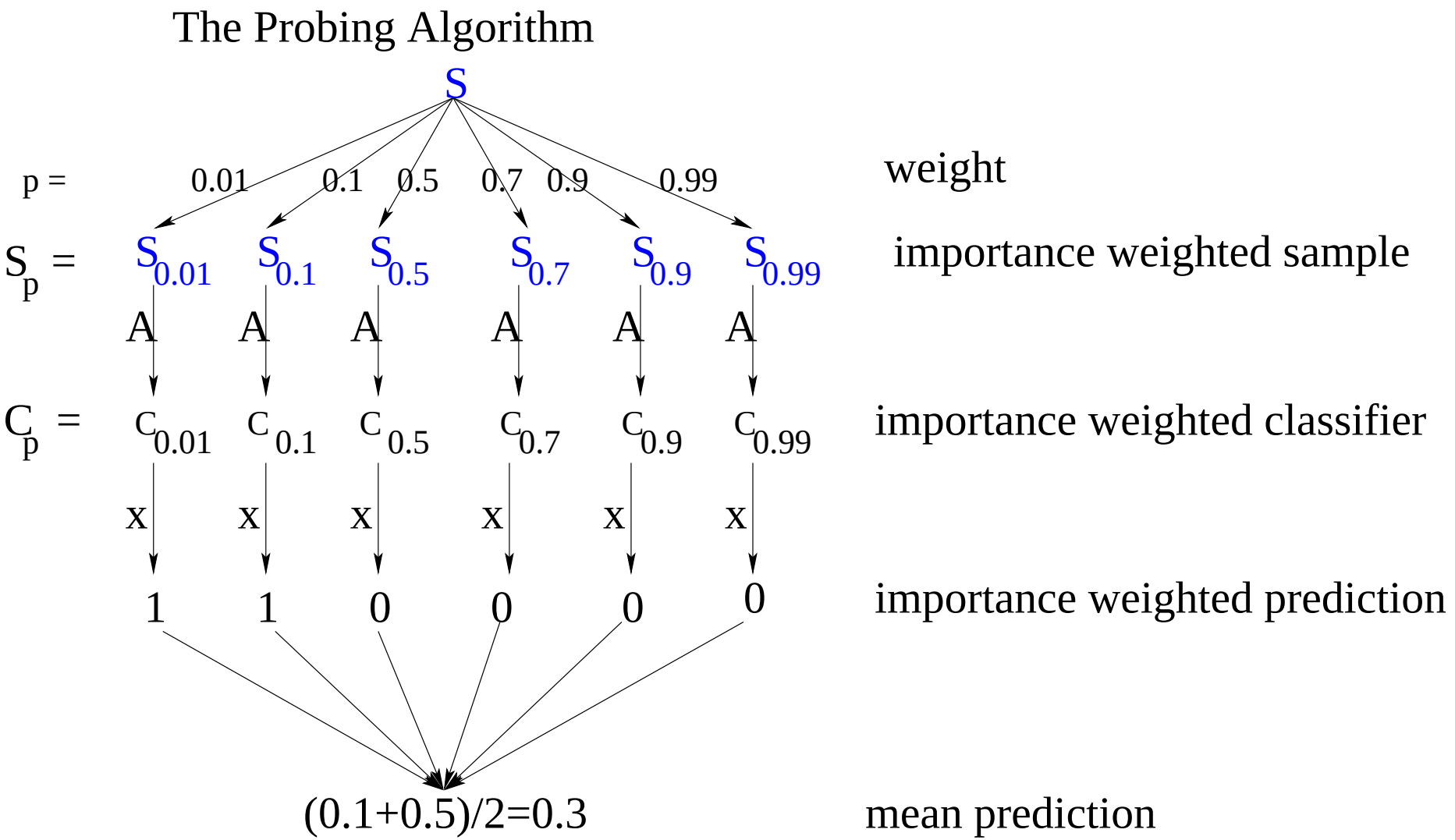
If $c$ perfect then, $c(x) = 1 \Rightarrow E_{y \sim D|x}[y] > t$

Proof:

$c(x) = 1 \Rightarrow E_{y \sim D|x} I(y \leq t)(t - y) < E_{y \sim D|x} I(y > t)(y - t)$

$\Rightarrow 0 < E_{y \sim D|x} I(y > t)(y - t) + E_{y \sim D|x} I(y \leq t)(y - t)$

$\Rightarrow 0 < E_{y \sim D|x}[y - t]$

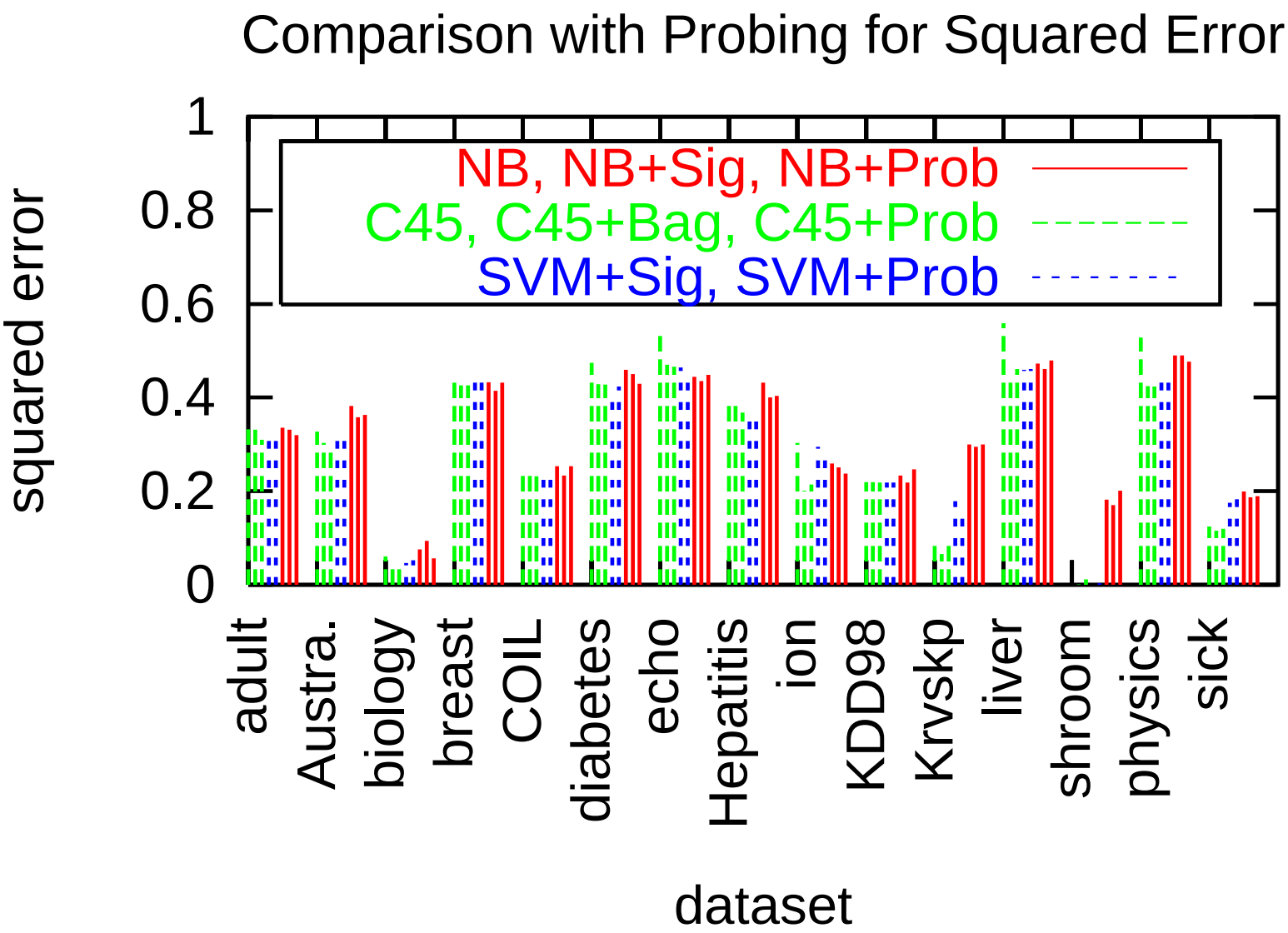

The Probing Algorithm
S
p =
0.01
0.1
0.5
0.7
0.9
0.99
weight
S_p =
S_0.01
S_0.1
S_0.5
S_0.7
S_0.9
S_0.99
importance weighted sample
A
C_p =
C_0.01
C_0.1
C_0.5
C_0.7
C_0.9
C_0.99
importance weighted classifier
x
1
1
0
0
0
0
importance weighted prediction
(0.1+0.5)/2=0.3
mean prediction

## The Probing Method: Details

1. How do you make classifier take weights? Costing Reduction

2. How do you deal with nonmonotonic predictions? Sort

3. How do you discretize on $t$? Uniform grid or on demand


Comparison with Probing for Squared Error
squared error
1
0.8
0.6
0.4
0.2
0
NB, NB+Sig, NB+Prob
C45, C45+Bag, C45+Prob
SVM+Sig, SVM+Prob
adult
Austra.
biology
breast
COIL
diabetes
echo
Hepatitis
ion
KDD98
Krvskp
liver
shroom
physics
sick
dataset

## The one classifier trick

We learn many classifiers in parallel. They *could* be one classifier.

1. Let $S = \cup_t \{(< x, t >, y, i) : (x, y, i) \in S_t\}$
2. Let $c = \text{Costing}(S, A)$
3. Let $c_t(x) = c(x, t)$

Good for practice? Unknown.

Handy for theory: we can think about drawing from induced distribution on $c$ by drawing from $(x, y) \sim D$ and $t \sim U(0, 1)$ to generate a sample from induced distribution $\text{Probing}(D)$.

## Probing Theory

Cute observation: $\text{Probing}_c(x) = \Pr_{t \sim U(0,1)}(c(x,t) = 1)$

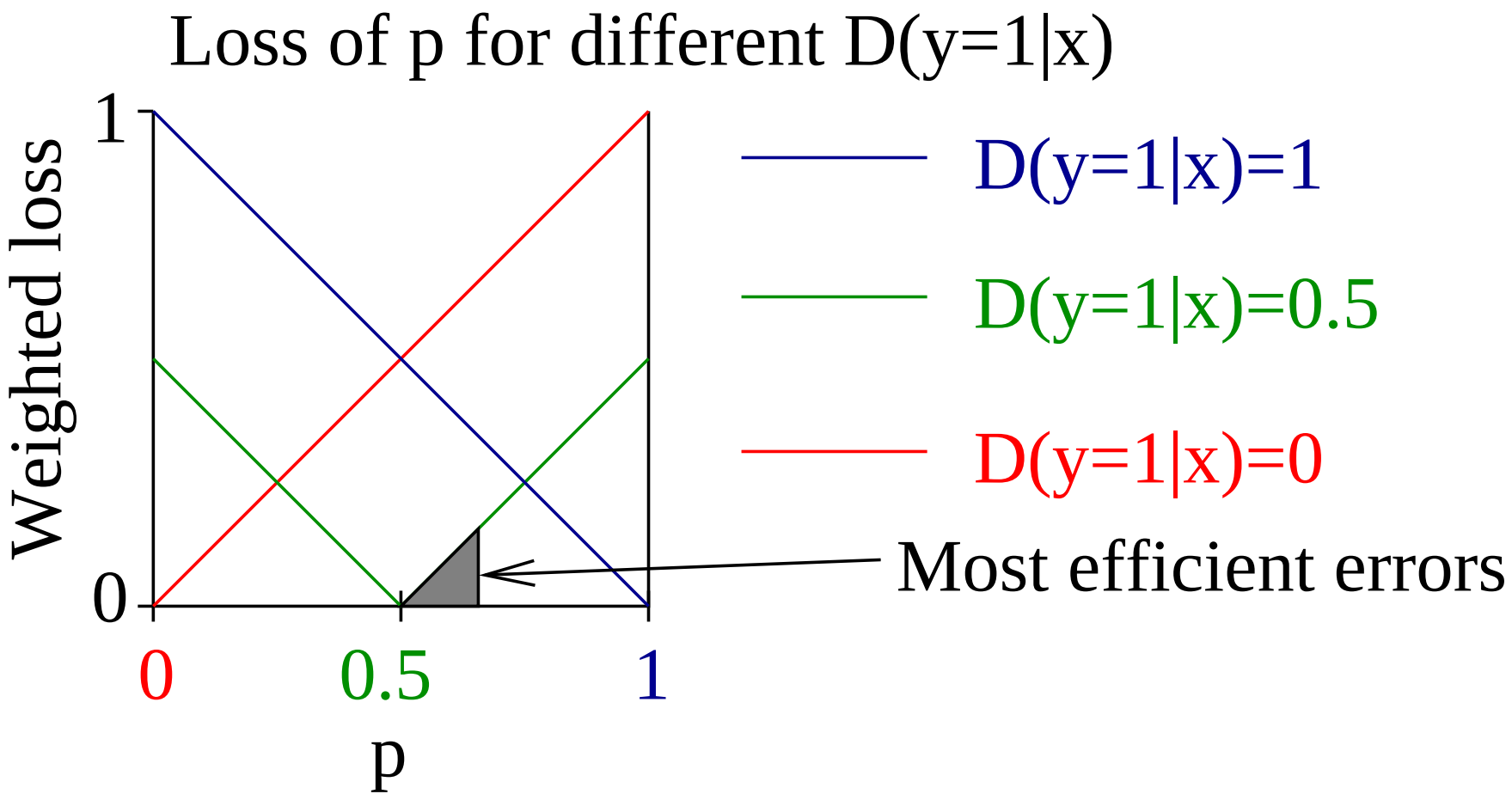
Theorem: (Probing Translation) For all $c : X \times [0,1] \to \{0,1\}$, for all $D$ on $X \times [0,1]$

$$E_{x,y \sim D}(E_{y \sim D|x}[y] - \text{Probing}_c(x))^2$$

$$\leq e(\text{Probing}(D), c) - \min_{c'} e(\text{Probing}(D), c')$$

... spooky. You don't know $E[y|x]$, yet minimizing $e(\text{Probing}(D), c)$ always implies good estimates of $E[y|x]$.

## The proof, pictorially



## The proof, mathematically

Expected importance $\leq \frac{1}{2}$ so:

$e(\text{Probing}(D), c) - \min_{c'} e(\text{Probing}(D), c')$

$= 2E_{t,(x,y)\sim D}|y - t|I(c(x,t) \neq y)$

$\quad - \min\{E_{t,(x,y)\sim D} I(y \leq t)(t - y), E_{t,(x,y)\sim D} I(y > t)(y - t)\}$

("2" comes from the distribution shift theorem)

$= 2E_{x,t}E_{y\sim D|x}|y - t|I(c(x,t) \neq y) -$

$\quad - \min\{E_{y\sim D|x} I(y \leq t)(t - y), E_{y\sim D|x} I(y > t)(y - t)\}$

## Proof, continued

For any $x, t$, either $c$ predicts perfectly (difference $= 0$) or not.

If not, difference $= |E_{y\sim D|x} I(y \leq t)(t-y) - E_{y\sim D|x} I(y > t)(y-t)|$

$= |E_{y\sim D|x} I(y \leq t)(t-y) + E_{y\sim D|x} I(y > t)(t-y)|$

$= |t - E[y|x]|$

$\Rightarrow$ cost of misclassification $= 2|t - E[y|x]|$.

## Proof II: Properties of most efficient error inducing method

How can $\epsilon$ binary errors maximize probability estimation errors?

1. Only classifications $c(x,t)$ on one side of $E[y|x]$ err. (otherwise sort = cancellation of errors = wasted binary errors)

2. Errors for $t$ closer to $E[y|x]$ are preferred over errors further from $E[y|x]$ (sort makes these equivalent, and the importance weighted loss is smaller for nearer errors)

$\Rightarrow$ deviation $\Delta$ requires at least importance weighted loss

$$\int_{E[y|x]}^{E[y|x]+\Delta} 2|t - E[y|x]|dt = \Delta^2$$

## An Modification: Quantile Regression

Goal: Given $q \in [0, 1]$, predict $\hat{y}$ such that $\Pr(y \geq \hat{y}) = q$

Algorithm: the same except $(x, y) \to (x, I(y > t), qI(y > t) + (1 - q)I(y < t))$

Theorem: (for the $q = 0.5$ median case)

$$E_{x,y\sim D}|y - \text{Quanting}_c(x)| - \min_{q(x)} E_{x,y\sim D}|y - q(x)|$$

$$\leq e(\text{Quanting}(D), c) - \min_{h'} e(\text{Quanting}(D), c')$$

Proof: Similar to Probing proof.

Quantile Prediction Performance
normalized quantile loss
1
0.8
0.6
0.4
0.2
0
linear
kernel
quanting/log. reg.
quanting/J48
adult 0.1
adult 0.5
adult 0.9
KDD98 0.1
KDD98 0.5
KDD98 0.9
Calif. 0.1
Calif. 0.5
Calif. 0.9
Boston 0.1
Boston 0.5
Boston 0.9
dataset

## Some Caveats

1. No bound holds for cross entropy. (Can't be done without extra assumptions/constraints.)

2. This approach can not bound *relative ranking* loss.

Given a Binary classifier, how can we solve

1. Importance Weighted Classification?

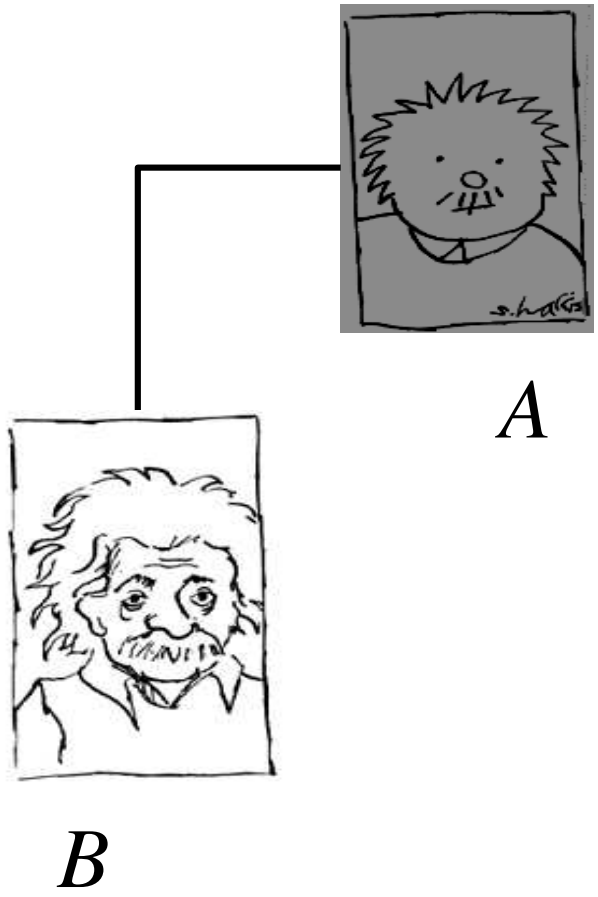
2. Regression?

3. Multiclass Classification?

4. Cost Sensitive Classification?

# Multiclass Reductions

Reductions turn a (black-box) learner for *A* into a learner for *B*.



*A*

*B*

## Online Learning

The number of mistakes we make is never more than *f* (number of mistakes made by the best expert so far)

## PAC Learning

Concept class *C*, hypothesis class *H*.

The learner needs to observe at most poly many i.i.d. examples to output a hypothesis *h* in *H* that has small error with high probability.

Algorithms: consistency

## Learning Reductions

If my oracle does well on its subproblems, I will do well on the original problem

**VC Learning** **Bayes Learning**

**Agnostic Learning**

No assumptions

Weaker (relative) statements

Always applicable

Strong Assumptions

Strong Statements

Weak applicability

## Multiclass Classification

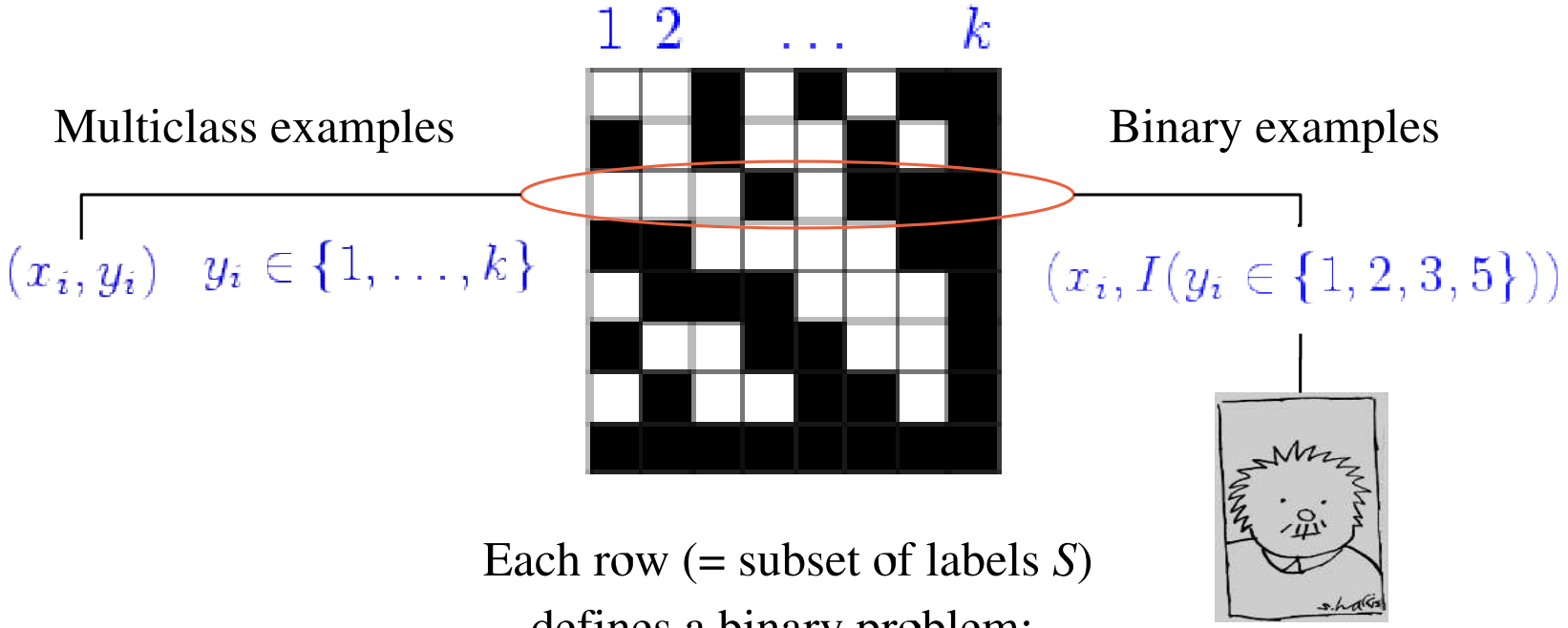
- Problem: A measure $D$ on $X \times \{1, ..., k\}$ where $X$ is an arbitrary space.

- Multiclass Classifier: $c_m : X \to \{1, ..., k\}$ = predictor

- Given $S = (X \times \{1, ..., k\})^*$ find multiclass classifier $c_m$ with small error rate:

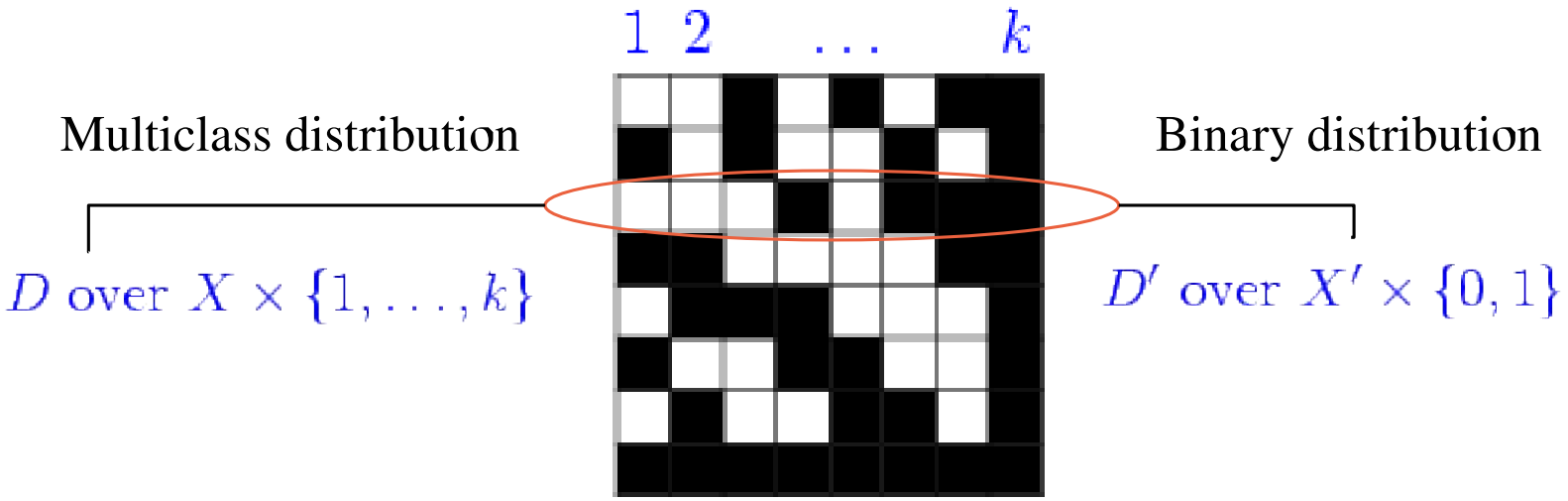
$$e(D, c_m) = \Pr_{(x,y)\sim D}(c_m(x) \neq y)$$

# ECOC Transformation




Each row (= subset of labels *S*)
defines a binary problem:
"Is the label in *S* or not?"

# ECOC Transformation

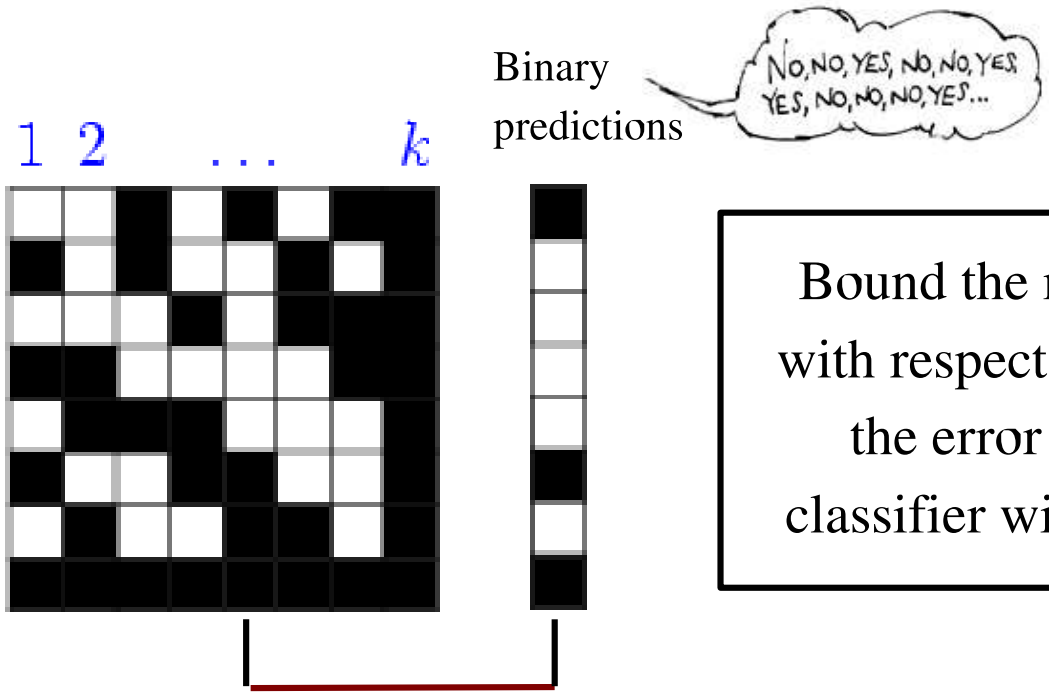



Draw $(x, y) \sim D$, a random row $S$, output $(\langle x, S \rangle, I(y \in S))$

# Error Transformation

How does performance on $D'$ imply performance on $D$ ?

# ECOC Prediction




Bound the multiclass error with respect to $D$ in terms of the error of the binary classifier with respect to $D'$.

# ECOC Analysis

Use one classifier trick $c(x, s) = c_s(x)$. Let ECOC($D_m$)= induced distribution on binary predictions.

Theorem: (ECOC transform) For all $k$, there exists code such that for all $c$, for all $D_m$ on $(x, y_m)$,

$$e(D_m, \text{ECOC}_c) \leq 4e(\text{ECOC}(D_m), c)$$

Proof: Exists code such that $< \frac{1}{4}$ binary errs $\Rightarrow$ no error

Intuition: two random bit vectors disagree in $\frac{1}{2}$ of locations so $\frac{1}{4}$ positions must be flipped to change which vector is nearest.

Exact example = rows of Hadamard matrix.

# Two little problems

1) Not interesting if the error rate of the oracle is > ¼ .

2) Inconsistency:

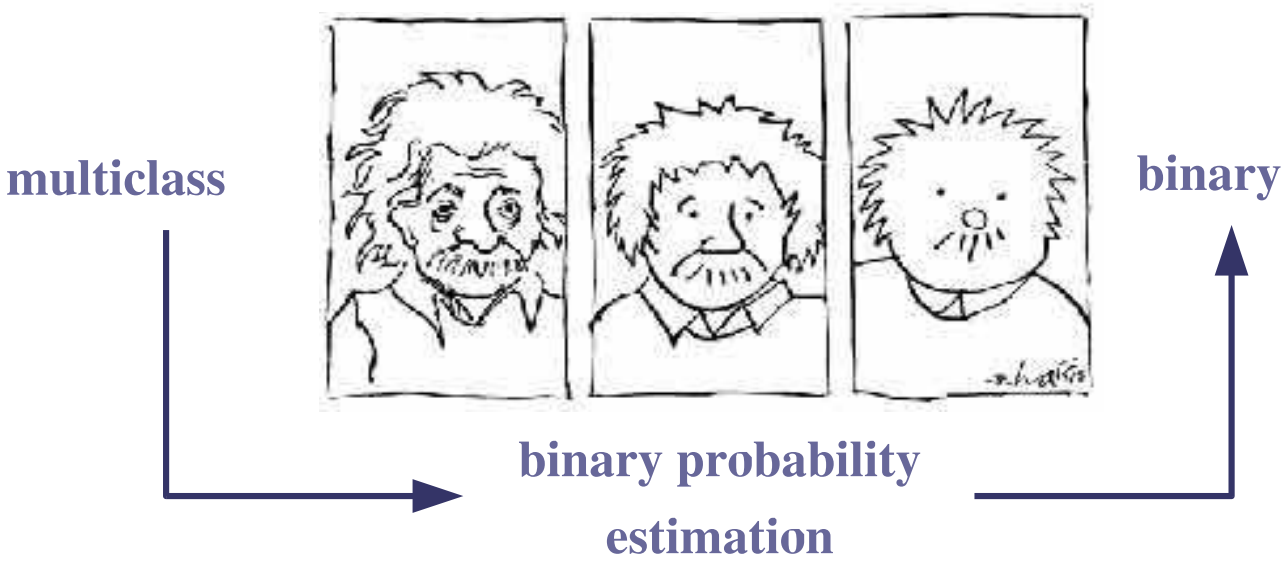
**ECOC Inconsistency:** There exists *D* such that for all codes,

ECOC with $c^* = \text{argmin}_c e(\text{ECOC}(D), c)$

fails to provide an optimal multiclass classifier:

$$e(D, \text{ECOC}_{c^*}) - \min_h e(D, h) > 0$$

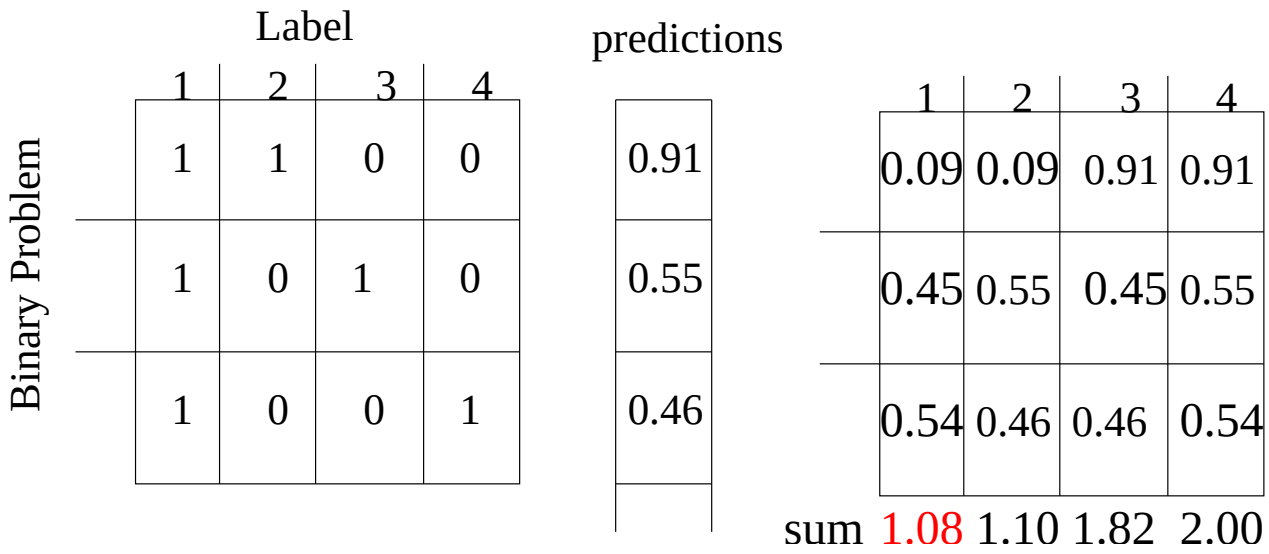
# PECOC: Probabilistic ECOC

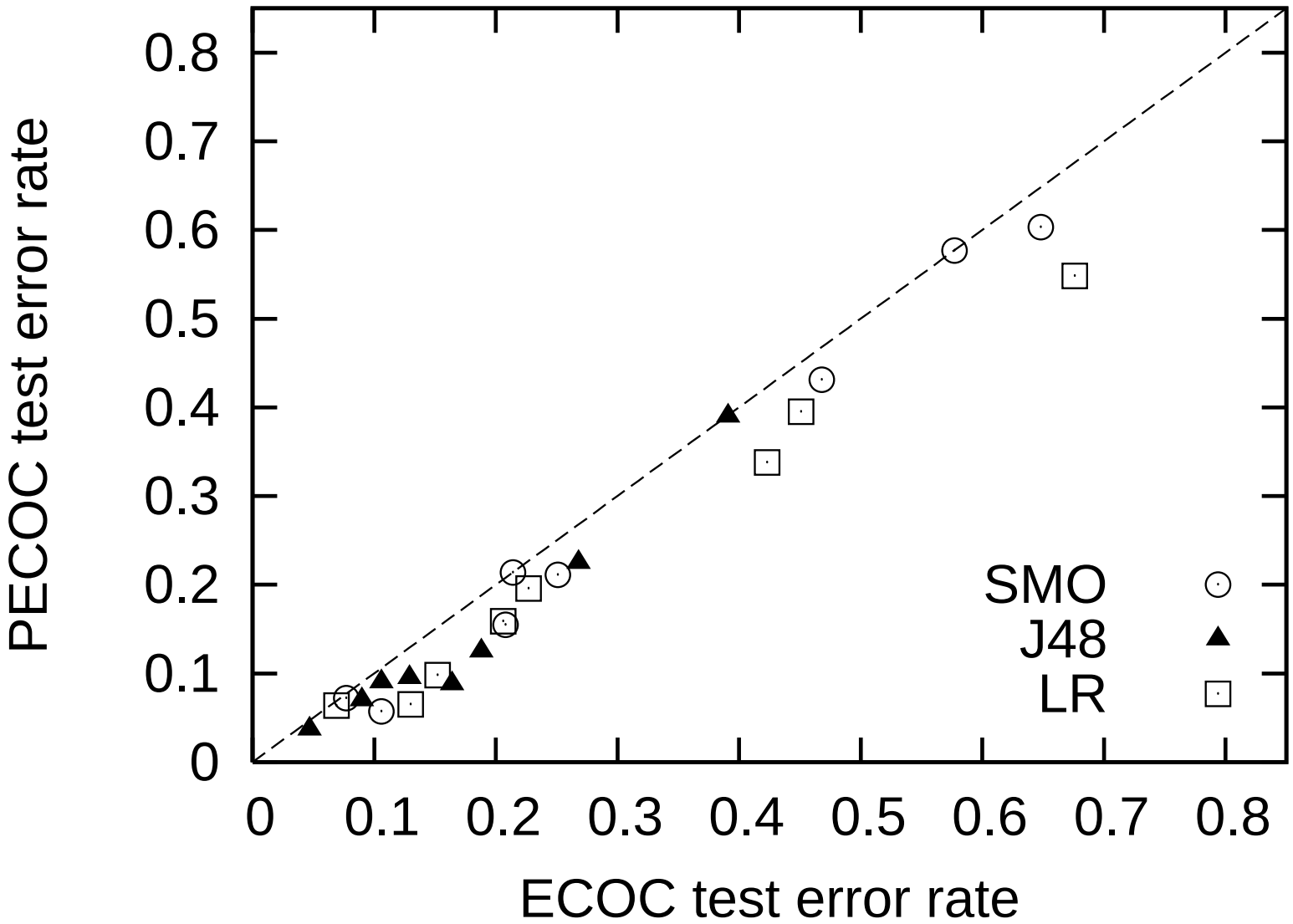


# Probabilistic Error Correcting Output Code

As ECOC, except use Probing to get probabilistic predictions.

To predict: Use $l_1$ closest label/codeword.




PECOC test error rate
ECOC test error rate
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
SMO
J48
LR

## PECOC Analysis

Use one classifier trick $c(x, s, p) = c_{sp}(x)$. Let PECOC($D_m$)= induced distribution on binary predictions.

Theorem: (PECOC transform) For all $k$, there exists code such that for all $c$, for all $D_m$ on $(x, y_m)$,

$$e(D_m, \text{PECOC}_c) - \min_{c'} e(D_m, c')$$

$$\leq 4\sqrt{e(\text{PECOC}(D_m), c) - \min_{c'} e(\text{PECOC}(D_m), c')}$$

- Binary consistency $\Rightarrow$ Multiclass consistency

- Binary error rate $= 0.25$ OK when minimum error rate $= 0.25$

# Proof

Pick code = subset of recursively defined Hadamard matrix.

$$
\begin{array}{llllllllllllllll}
 & & & & & & & & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 & & & & & & & & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
 & & & & & & & & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
 & & & & & & & & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
 & & & & & & & & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
 & & & & & & & & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
 & & & & & & & & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\
 & & & & & & & & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
 & & & & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
 & & & & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
 & & & & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\
 & & & & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
 & & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\
 & & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
1) & 1 & 0 & 2) & 1 & 0 & 0 & 1 & 3) & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 4) & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1
\end{array}
$$

If $b$ = number of binary problems, distance between codewords $= \frac{b}{2}$

For all labels $l$,

$$\sum_{b \in \text{Binary Problems}} \text{Probing}_b(\text{set containing } l|x)$$

$$= \sum_{b \in \text{Binary Problems}} \left[ \sum_{l' \in \text{set containing l}} D_m(l'|x) \right]$$

(Assuming perfect classifiers)

$$= b(D_m(l|x) + \frac{1}{2} \sum_{l' \neq l} D_m(l'|x)) = b\frac{D_m(l|x) + 1}{2}$$

## Proof: Analyzing errors

From Probing analysis,

$$E_{x,y\sim D}|D(1|x) - \text{Probing}_c(x)|$$

$$\leq \sqrt{e(\text{Probing}(D), c) - \min_{c'} e(\text{Probing}(D), c')}$$

Let $b$ = number of binary problems.

Most efficient method to disturb $l_1$ sum by $\epsilon b$ is for each call to probing to err by $|D(1|x) - \text{Probing}(x)| = \epsilon$. (since $\sqrt{x}$ is convex)

Changing $l_1$ sum by $b\epsilon \Rightarrow$ choosing class with error rate at most $4\epsilon$ worse than minimum.

## An Extension

You can also do probabilistic multiclass prediction.

$$\sum_{b \in \text{Binary Problems}} \widehat{p}_b = b \frac{D_m(l|x) + 1}{2}$$

$$\Rightarrow D_m(l|x) = \frac{2 \sum_{b \in \text{Binary Problems}} \widehat{p}_b}{b} - 1$$

... and this turns out to be stable.

Theorem: (PECOC' Translation) For all $c : X \times [0,1] \to \{0,1\}$, for all $D_m$ on $X \times \{1, ..., k\}$

$$E_{x,y \sim D}(D_m(y|x) - \text{PECOC'}_c(x))^2$$

$$\leq 4(e(\text{PECOC}(D_m), c) - \min_{c'} e(\text{PECOC}(D_m), c'))$$

Given a Binary classifier, how can we solve

1. Importance Weighted Classification?

2. Regression?

3. Multiclass Classification?

4. Cost Sensitive Classification?

## Cost Sensitive Classification

- Problem: A measure $D_{cs}$ on $X \times [0, \infty)^k$ where $X$ is an arbitrary space.

- Multiclass Classifier: $c_m : X \to \{1, ..., k\}$ = predictor

- Given $S = (X \times [0, \infty)^k)^*$ find classifier with small expected loss

$$e_{cs}(D_{cs}, c_m) = E_{(x,\vec{\ell}) \sim D_{cs}}[\ell_{c_m(x)}]$$

# Sensitive Error Correcting Output Code

SECOC = cost sensitive $\Rightarrow$ importance weighted classification reduction, in two parts. Uses a code matrix $M$:

| Subset \ Label | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| | 1 | 1 | 0 | 0 |
| | 1 | 0 | 1 | 0 |
| | 1 | 0 | 0 | 1 |

# SECOC, the training algorithm

**SECOC-Train** (cost sensitive examples $S$, importance weighted learning algorithm $A$)

1. For each subset $s$ defined by the rows of $M$:
   (a) For $(x, \vec{\ell}) \in S$, let $|\vec{\ell}| = \sum_y \ell_y$ and $\ell_s = \sum_{y \in s} \ell_y$.
   (b) For random $t$ in $[0, 1]$:
   Let $c_{st} = A(\{(x,\ I(\ell_s \geq t|\vec{\ell}|), |\ell_s - |\vec{\ell}|t|) : (x, \vec{\ell}) \in S\})$.
2. return $\{\ell_{st}\}$

SECOC, the prediction algorithm

**SECOC-Predict** (classifiers $\{c_{st}\}$, example $x \in X$)

return $\min_y E_s E_t \left[ I(y \in s) c_{st}(x) + I(y \notin s)(1 - c_{st}(x)) \right]$

# SECOC Analysis

Use the one classifier trick + Costing. Let

$$S'_{st} = \{((x, s, t), y, i) : (x, y, i) \in S_{st}\}$$

Then train to learn $c = A(\cup_{st} S'_{st})$ and define $c_{st}(x) = c(x, s, t)$

Theorem: (SECOC transform) For all $k$, there exists code such that for all $c$, for all $D_{cs}$ on $(x, \vec{\ell})$,

$$e_{cs}(D_{cs}, \mathsf{SECOC}_c) - \min_{c'} e_{cs}(D_{cs}, c')$$

$$\leq 4\sqrt{(e(\mathsf{SECOC}(D_{cs}), c) - \min_{c'} e(\mathsf{SECOC}(D_{cs}), c'))E_{(x,\vec{\ell})\sim D_{cs}}|\vec{\ell}|}$$

Proof (sketch only, much like PECOC)

1. $E_t\left[I(y \in s)c_{st}(x) + I(y \notin s)(1 - c_{st}(x))\right] = \frac{E_{\vec{\ell} \sim D|x}[\ell_s]}{E_{\vec{\ell} \sim D|x}\left[|\vec{\ell}|\right]}$ when classifiers optimal.

2. $E_s \frac{E_{\vec{\ell} \sim D|x}[\ell_s]}{E_{\vec{\ell} \sim D|x}\left[|\vec{\ell}|\right]} = \frac{\frac{E_{\vec{\ell} \sim D|x}[\ell_y]}{E_{\vec{\ell} \sim D|x}\left[|\vec{\ell}|\right]} + 1}{2}$ when classifiers optimal.

3. Optimal method for adversary to cause loss without incurring importance weighted regret = small error $t$'s for each $s$.

4. Cost of erring linear in $t \Rightarrow$ average regret growth quadratic.

## Another Bonus

Corollary: Soft Prediction

$$\Rightarrow E_{x \sim D_{cs}} \left( \mathsf{Predict}(c, x, y) E_{\vec{\ell}' \sim D|x} \left[ |\vec{\ell}'| \right] - E_{\vec{\ell}' \sim D|x} \left[ \ell'_y \right] \right)^2$$

$$\leq 4(e(\mathsf{SECOC}(D_{cs}), c) - \min_{c'} e(\mathsf{SECOC}(D_{cs}), c')) E_{(x, \vec{\ell}) \sim D_{cs}} |\vec{\ell}|$$

## Some Things to Think About

Experiments: Not done yet. How well does this work in practice?

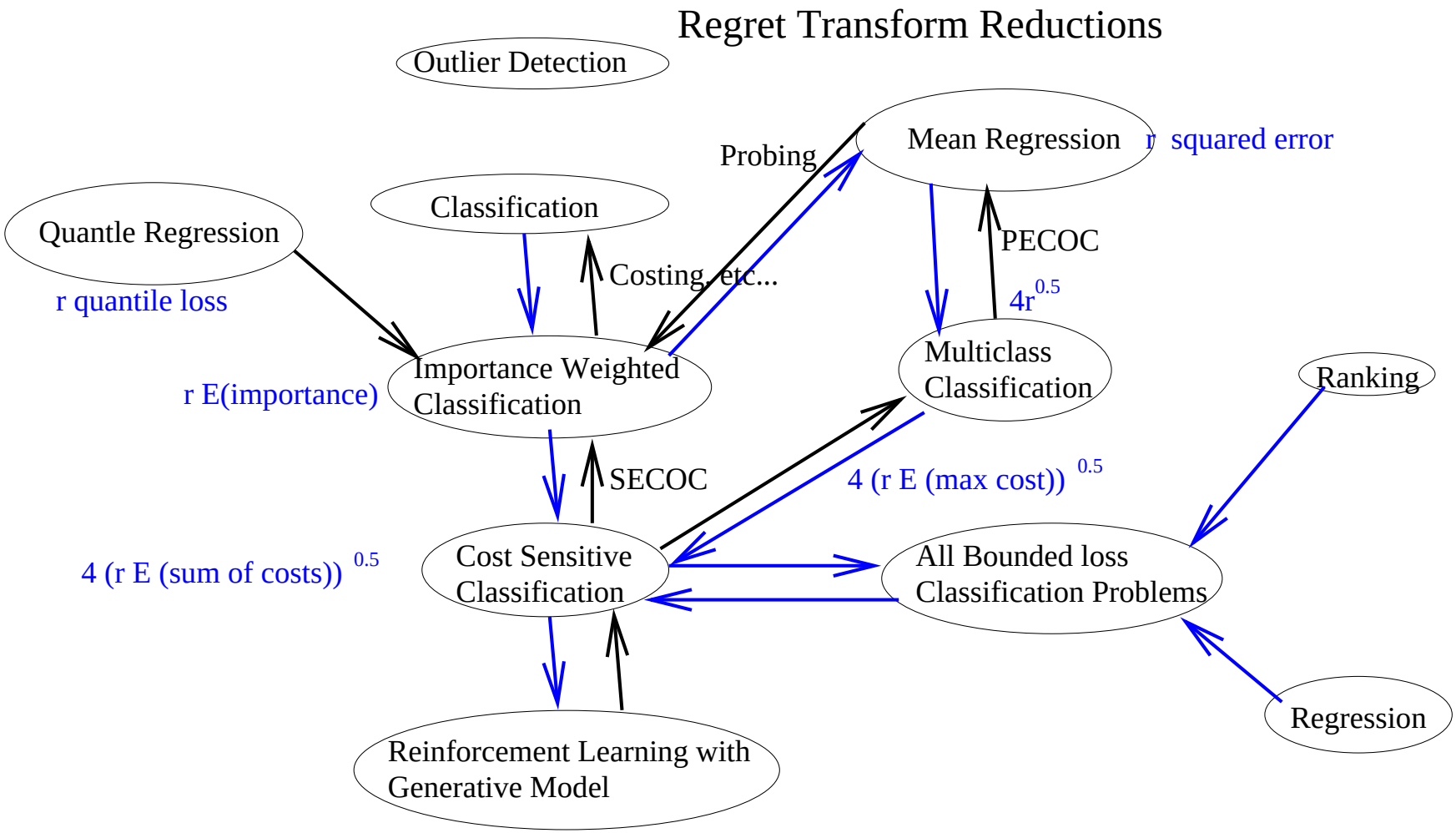
Which code do you use in practice?

Shannon $\Rightarrow$ a random code of size $O(\log k)$ is near optimal for above theory.

# Final Thoughts

Formal study of learning reductions is relatively new.

- $\Rightarrow$ The limits of the possible are not entirely known.
- $\Rightarrow$ You-a-fellow-researcher could easily have important insights. New since MLSS 2005 in Chicago:
  - Probing does arbitrary squared error regression.
  - Quanting Reduction
  - Better cost sensitive reduction analysis. (Not detailed here.)


Regret Transform Reductions
Outlier Detection
Mean Regression
r squared error
Probing
Classification
Quantle Regression
r quantile loss
Costing, etc...
PECOC
4r^0.5
Multiclass Classification
Importance Weighted Classification
r E(importance)
Ranking
SECOC
4 (r E (max cost)) ^0.5
4 (r E (sum of costs)) ^0.5
Cost Sensitive Classification
All Bounded loss Classification Problems
Regression
Reinforcement Learning with Generative Model

## Related Reading

[ECOC] T. Dietterich and G. Bakiri, "Solving Multiclass Classification via Error-Correcting Output Codes", JAIR 2:263-296, 1995.

[Boosting] Y. Freund and R. Schapire, "A Decision-Theoretic Generalization of Online Learning and an Application to Boosting", JCSS 55(1) 119-39, 1997.

[ECOC analysis] V. Guruswami and A. Sahai, "Multiclass Learning, boosting, and error-correcting codes", COLT 145-55, 1999.

[ECOC variant] E. Allwein, R. Schapire, and Y. Singer, "Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers", JMLR, 1:113-41, 2000.

[Costing] B. Zadrozny, J. Langford, and N. Abe. Cost-Sensitive L earning by Cost-Proportionate Example Weighting. ICDM 435–42, 2003.

[Probing] J. Langford and B. Zadrozny, "Estimating Class Membership Probabilities Using Classifier Learners", AISTAT 2005.

[PECOC & SECOC] J. Langford and A. Beygelzimer, "Sensitive Error Correcting Output Codes", COLT 2005.

[Quanting] J. Langford, R. Oliveira, and B. Zadrozny, "Predicting Conditional Quantiles via Reduction to Classification", UAI 2006.