Machine Learning Reductions

A Taxonomy of Learning Theory

1. (Some) online learning

2. Reductions

3. Algorithmic complexity?

4. Data Constrained $\Rightarrow$ (other) online learning

5. I.I.D. Data $\Rightarrow$ learning verifiability

6. I.I.D. Data $+$ Data Constrained $\Rightarrow$ PAC-learning

"I am not a predictor." - President Bush

Prediction is hard. Very hard. In fact: worst case impossible

... yet sometimes we succeed.

How can we analyze something which is worst case impossible?

Use reductions

Given: Classification algorithm A

Construct: Learning algorithm R^A for “cost sensitive classification”, “reinforcement learning”, “regression”, etc...

Construct = Reduction

More formally:

1. Define learning domain.
2. Learning problems exist within a learning domain.
3. Learning algorithms solve learning problems within a domain.
4. Reductions make learning algorithms solve learning problems within new domains.

Learning Domains

Learning Domain = $L = (K, Y, l)$

K = information about *what* should be predicted.

Y = the space of predictions

$l : K \times Y \to [0, \infty)$ = "loss" function

Learning Problem

Learning problem $=$

- (K, Y, l) Learning Domain

-X $=$ “feature space” $=$ the thing you predict from

-D a distribution over $X \times K$

Features of Learning Problems

1. Learning problem D + predictor $h : X \to Y$ implies loss rate,

$$h_D = E_{(x,k)\sim D} l(k, h(x))$$

2. No explicit notion of time.

Learning Problem examples

1. Domain = classification, $K = Y =$ snow or not, $X =$ weather report

2. Domain = importance classification, $X =$ past history, $K =$ Fraud or not (and cost of fraud or unnecessary study).

3. Domain = Cost sensitive classification, $X =$ hour of day, $K =$ cost of taking various paths, $Y =$ path taken.

Learning Domains, Examples

L	K	Y	$l(k, y)$
Classification			
Importance Classification			
ℓ-class Classification			
Ranking			
n-of-ℓ Classification			
Cost Sensitive Classification			
Regression			

Learning Domains, Examples

L	K	Y	$l(k, y)$
Classification	$\{0, 1\}$	$\{0, 1\}$	$I(k \neq y)$
Importance Classification	$\{0, 1\} \times [0, \infty)$	$\{0, 1\}$	$k_2 I(k_1 \neq y)$
ℓ-class Classification	$\{1, ..., \ell\}$	$\{1, ..., \ell\}$	$I(k \neq y)$
Ranking	$\{1, ..., \ell\}$	$\{1, ..., \ell\}$	$\|k - y\|$
n-of-ℓ Classification	$2^{\{1, ..., \ell\}}$	$\{1, ..., \ell\}$	$I(y \notin k)$
Cost Sensitive Classification	$[0, \infty)^\ell$	$\{0, ..., \ell\}$	k_y
Regression	$(-\infty, \infty)$	$(-\infty, \infty)$	$(k - y)^2$

More formally:

1. Define learning domain.

2. Learning problems exist within a learning domain.

3. Learning algorithms solve learning problems within a domain.

4. Reductions make learning algorithms solve learning problems within new domains.

Learning Algorithm = Domain solver

Examples:

1. Support Vector Machines (Domain = Classification)
2. EM (Domain = Density estimation)
3. Neural Networks (Domain = Classification or Regression)

Reduction $R(A) = L'$ solver given L solver, A

Reduction satisfies one property:

1. (Error limitation) If L solver does well on derived problems then reduction does well on original problem.

Derived Problem

Let

$$D_{S'}(x,k) = \frac{|\{(x,k) : (x,k) \in S\}|}{|S|}$$

Derived Problem:

$$D(x,k) = E_{S' \sim D'} D_{S'}(x,k)$$

Reduction $R(A) = L'$ solver given L solver, A

Reduction satisfies one property:

1. (Error limitation) If for all created sets S, $A(S)$ produces a predictor h with $h_D \leq \epsilon$, then $R(A, S')$ has error rate less than $g(\epsilon)$ where $g(0) = 0$ and $g(z) \geq g(z')$ for $z \geq z'$.

Let R_{12} be a reduction from L_1 to L_2 with error transform g_{12}.

Let R_{23} be a reduction from L_2 to L_3 with error transform g_{23}.

Theorem: $R_{12} \circ R_{23}$ is a reduction from L_1 to L_3 with error transform $g_{12} \circ g_{23}$.

Proof:

1. Everything has the right type.

2. Error bounds propagate.

The rest

1. Importance Classification to Classification

2. An overview of reduction results

Domain Comparison

L	K	Y	$l(k,y)$
Binary Classification	$\{0,1\}$	$\{0,1\}$	$I(k \neq y)$
Importance Classification	$k_2 I(k_1 \neq y)$	$\{0,1\}$	$k_2 I(k_1 \neq y)$

Reduction Construction

Given:

1. $(x, y, i)^m$

2. classification algorithm, $A : (X \times \{0,1\}) \to (X \to \{0,1\})$

construct:

1. $A' : (X \times \{0,1\} \times [0, \infty)) \to (X \to \{0,1\})$

Basic theory: A Folk Theorem

Let $D'(x, y, i) \propto iD(x, y, i)$

Let $N = E_{x,y,i \sim D} i$

Theorem:

For all c, $E_{x,y,i \sim D'} I(c(x) \neq y) = \frac{1}{N} E_{x,y,i \sim D} i I(c(x) \neq y)$

$\Rightarrow$ if we transform examples to be from D', and apply A, we have a learning reduction.

Attempt #1: Resampling

Resample(A, $(x, y, i)^m$)

1. $S = m$ times: draw (x, y) proportional to i

2. return $A(S)$

Basic problem with resampling: samples in S not drawn i.i.d. from correct distribution.

$\Rightarrow$ Learning algorithms fail

Attempt #2: Rejection Sample

Rejection_Sample(A, $(x, y, i)^m$)

1. $S =$ for each (x, y, i) if Rand($\max_{(x,y,i)\in S} i$)$<$ i then include (x, y)

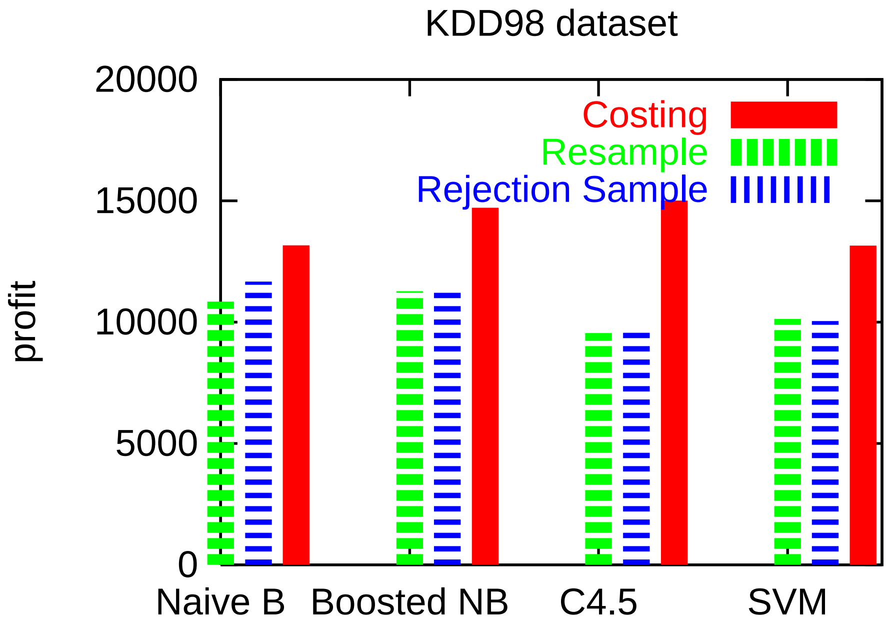
2. $A(S)$

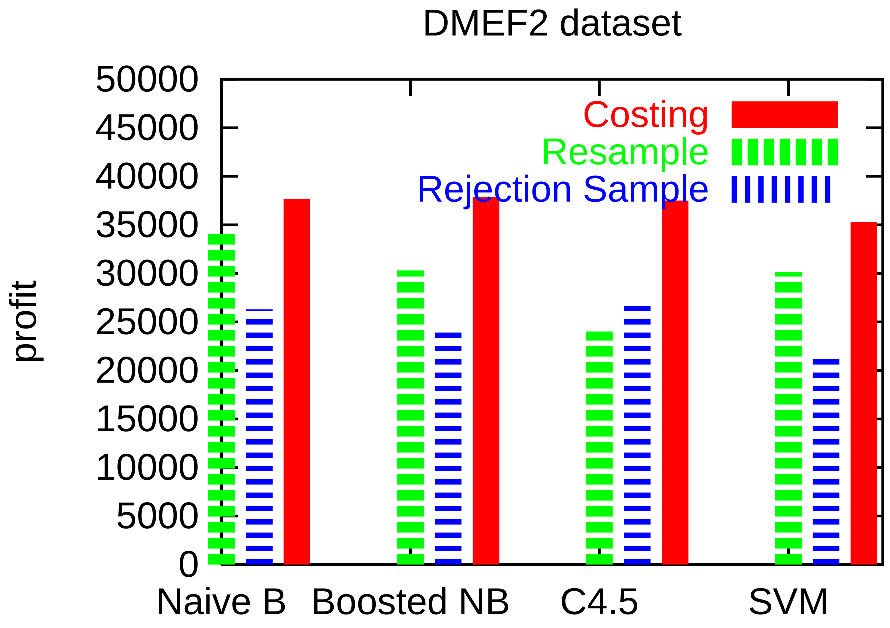
The right distribution, but we throw away too many samples.

Attempt #3: "Costing"

Costing(A, $(x, y, i)^m$)

1. For $t = 1$ to 200
2. $c_t =$Rejection_Sample(A, $(x, y, i)^m$)
3. Return $c(x) = I(\frac{1}{200}\sum_{t=1}^{200} c_t(x) > 0.5)$


KDD98 dataset
profit
20000
15000
10000
5000
0
Costing
Resample
Rejection Sample
Naive B
Boosted NB
C4.5
SVM


DMEF2 dataset
profit
50000
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
Costing
Resample
Rejection Sample
Naive B
Boosted NB
C4.5
SVM

Learning Preserving: Definition

A reduction R from L' to L is learning preserving if there exists a reduction R' from L to L' such that for all oracle learning algorithms A and all datasets S:

$$R'(R(A), S) = A(S)$$

Theorem: Costing is learning preserving

Proof: Let $R'(A', S) = A'(\{(x, y, 1) : (x, y) \in S\})$.

Rejection_Sample(A, S')

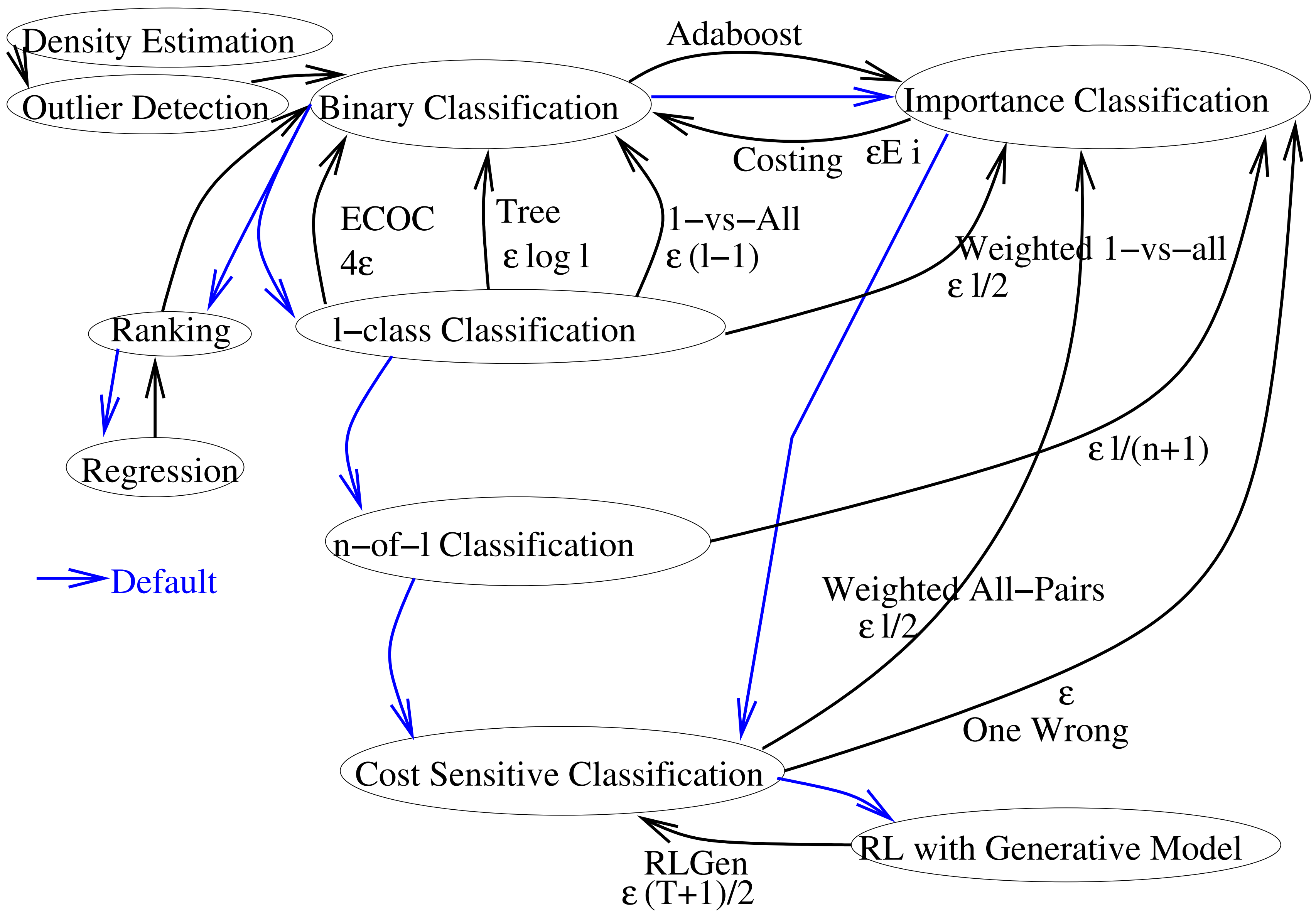
$$= A(\{(x, y) : (x, y, i) \in S' \text{ and } \mathrm{Rand}(\max_{(x,y,i)\in S} i) < i\})$$

$\max i = \min i = 1 \Rightarrow$ no examples rejected.

$\Rightarrow$ Rejection_Sample learning preserving

Costing $=$ majority vote of Rejection_Sample

$\Rightarrow$ Costing learning preserving.



Density Estimation
Outlier Detection
Binary Classification
Adaboost
Importance Classification
Costing
εE i
ECOC
4ε
Tree
ε log l
1−vs−All
ε (l−1)
Weighted 1−vs−all
ε l/2
Ranking
l−class Classification
Regression
ε l/(n+1)
n−of−l Classification
Default
Weighted All−Pairs
ε l/2
ε
One Wrong
Cost Sensitive Classification
RL with Generative Model
RLGen
ε (T+1)/2

A Table of Reduction Results

	Name	$g(\epsilon)$	Time	calls	IID?	Learning?
Class. to IC	Boosting	$e^{-2T(\frac{1}{2}-\epsilon)^2}$	$O(T)$	$O(T)$	Yes	??
IC to Class.	Costing	ϵEi	$O(1)$	$O(1)$	Yes	Yes
ℓ-class to C	ECOC	4ϵ	$O(1)$	$O(1)$	Yes	??
ℓ-class to C	1 vs all	$\epsilon(\ell - 1)$	$O(1)$	$O(1)$	Yes	??
ℓ-class to C	Tree	$\epsilon \log_2 \ell$	$O(1)$	$O(1)$	Yes	Yes
ℓ-class to IC	Wghtd 1vsAll	$\epsilon \frac{\ell}{2}$	$O(1)$	$O(1)$	Yes	??

C = Binary Classification

IC = Importance weighted Binary classification

IID? = Does the reduction preserve IID sampling?

Learning? = Is the reduction learning preserving?