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
Learning Domains

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

\( K \) = information about what should be predicted.

\( Y \) = the space of predictions

\( l : K \times Y \rightarrow [0, \infty) \) = “loss” function

Learning problem \( L, X, D = \)

- \( L = (K, Y, l) \) Learning Domain

- \( X \) = “feature space” = the thing you predict from

- \( D \) a distribution over \( X \times K \)
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

<table>
<thead>
<tr>
<th>$L$</th>
<th>$K$</th>
<th>$Y$</th>
<th>$l(k, y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification</td>
<td>${0, 1}$</td>
<td>${0, 1}$</td>
<td>$I(k \neq y)$</td>
</tr>
<tr>
<td>Importance Classification</td>
<td>${0, 1} \times [0, \infty)$</td>
<td>${0, 1}$</td>
<td>$k_2 I(k_1 \neq y)$</td>
</tr>
<tr>
<td>$\ell$-class Classification</td>
<td>${1, \ldots, \ell}$</td>
<td>${1, \ldots, \ell}$</td>
<td>$I(k \neq y)$</td>
</tr>
<tr>
<td>Ranking</td>
<td>${1, \ldots, \ell}$</td>
<td>${1, \ldots, \ell}$</td>
<td>$</td>
</tr>
<tr>
<td>$n$-of-$\ell$ Classification</td>
<td>$2^{{1, \ldots, \ell}}$</td>
<td>${1, \ldots, \ell}$</td>
<td>$I(y \notin k)$</td>
</tr>
<tr>
<td>Cost Sensitive Classification</td>
<td>$[0, \infty)^\ell$</td>
<td>${0, \ldots, \ell}$</td>
<td>$k_y$</td>
</tr>
<tr>
<td>Regression</td>
<td>$(-\infty, \infty)$</td>
<td>$(-\infty, \infty)$</td>
<td>$(k - y)^2$</td>
</tr>
</tbody>
</table>
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 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

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

Derived Problem:

\[
D(x, k) = E_{S' \sim D'} D_{S'}(x, k)
\]
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.
Import C. to Classification

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

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

Folk Theorem:

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

⇒ if we transform examples to be from $D'$, and apply $A$, we have a learning reduction.
IC to C 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
IC to C Attempt #2: Rejection Sample

Rejection\_Sample(\(A, (x, y, i)^m\))

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

2. \(A(S)\)

The right distribution, but we throw away too many samples.
IC to C Attempt #3: “Costing”

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

1. For $t = 1$ to $200$

2. $c_t = \text{Rejection Sample}(A, (x, y, i)^m)$

3. Return $c(x) = I\left(\frac{1}{200} \sum_{t=1}^{200} c_t(x) > 0.5\right)$
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 $R'(A', S) = A'((x, y, 1) : (x, y) \in S)$.

$\text{Rejection\_Sample}(A, S')$

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

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

$\Rightarrow \text{Rejection\_Sample learning preserving}$

Costing = majority vote of Rejection\_Sample

$\Rightarrow$ Costing learning preserving.
### A Table of Reduction Results

<table>
<thead>
<tr>
<th>Class. to IC</th>
<th>Boosting</th>
<th>$g(\epsilon) = e^{-2T(\frac{1}{2}-\epsilon)^2}$</th>
<th>$O(T)$</th>
<th>$O(T)$</th>
<th>Yes</th>
<th>??</th>
</tr>
</thead>
<tbody>
<tr>
<td>IC to Class.</td>
<td>Costing</td>
<td>$\epsilon Ei$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>$\ell$-class to C</td>
<td>ECOC</td>
<td>$4\epsilon$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
<td>Yes</td>
<td>??</td>
</tr>
<tr>
<td>$\ell$-class to C</td>
<td>1 vs all</td>
<td>$\epsilon (\ell - 1)$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
<td>Yes</td>
<td>??</td>
</tr>
<tr>
<td>$\ell$-class to C</td>
<td>Tree</td>
<td>$\epsilon \log_2 \ell$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>$\ell$-class to IC</td>
<td>Wghtd 1vsAll</td>
<td>$\epsilon \frac{\ell}{2}$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
<td>Yes</td>
<td>??</td>
</tr>
</tbody>
</table>

**C = Binary Classification**

**IC = Importance weighted Binary classification**

**IID? = Does the reduction preserve IID sampling?**

**Learning? = Is the reduction learning preserving?**