Machine Learning Reductions Tutorial

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

Yahoo Research

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

MLSS 2006, Taiwan, July 24
Scenario 1

You work for a charity as a spam optimizer.

Question: Who should you bother to ask money from?

“Oh look, this is binary classification. I’ll just apply a decision tree to predict who provides money.”

Result: $0.00 income on test set.

Oops: Mailing everyone ⇒ negative return. You are fired.
Scenario 2

You work for a doctor predicting “cancer or not” given symptoms.

You choose to use a support vector machine.

But the doctor doesn’t want a decision—a probability is preferred.

...so you return the “margin” as a probability.

Your probabilities are always near 0.5. You are fired.
Where did the Hollywood ending go?

Basic difficulty: World’s problem is not problem solved by algorithm.

Solutions:

1. Design new algorithms! (The research employment act.)

2. Discover how to reuse old algorithms.

Learning reductions are the mathematics of #2.

Basic question: Can #2 do everything that #1 can?
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 $R^A + \text{data}$. 
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: $c : X \rightarrow \{0, 1\} = \text{predictor}$

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

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

$$e_w(D, c) = E_{(x, y, i) \sim D}[i I(c(x) \neq y)]$$
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: $c(x) = \text{majority}(|c_1(x), \ldots, c_{10}(x)|)$
Costing+classifier applied to the KDD-98 dataset

championship winner

"profit"

Naive B.  B.N.B.  C4.5  SVM

classification algorithm
Costing+classifier applied to the DMEF2 dataset

"profit"

classification algorithm

Naive B.  B.N.B.  C4.5  SVM
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: $h : X \rightarrow [0, 1] = \text{predictor}$

- 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

\[ S_p = \]

\[ p = 0.01 \quad 0.1 \quad 0.5 \quad 0.7 \quad 0.9 \quad 0.99 \]

\[ S_{0.01} \quad S_{0.1} \quad S_{0.5} \quad S_{0.7} \quad S_{0.9} \quad S_{0.99} \]

weight

importance weighted sample

\[ C_p = \]

\[ c_{0.01} \quad c_{0.1} \quad c_{0.5} \quad c_{0.7} \quad c_{0.9} \quad c_{0.99} \]

importance weighted classifier

\[ (0.1+0.5)/2=0.3 \]

importance weighted prediction

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

- NB, NB+Sig, NB+Prob
- C45, C45+Bag, C45+Prob
- SVM+Sig, SVM+Prob
The one classifier trick

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

1. Let \( S = \bigcup_t \{ (\langle x, t \rangle, y, i) : (x, y, i) \in S_t \} \)

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 \( X \times [0, 1] \)

\[
E_{x,y \sim D}(y - \text{Probing}_c(x))^2 \\
\leq e(\text{Probing}(D), c) - \min_{c'} e(\text{Probing}(D), c')
\]

... 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

Loss of $p$ for different $D(y=1|x)$

- $D(y=1|x)=1$
- $D(y=1|x)=0.5$
- $D(y=1|x)=0$

Most efficient errors
The proof, mathematically

Expected importance \( \leq \frac{1}{2} \) so:

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

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

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

(“2” comes from the distribution shift theorem)

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

\[
- \min\{E_{y \sim D|x}I(y \leq t)(t - y), E_{y \sim D|x}I(y > t)(y - t)\}
\]
Proof, continued

For any $x, t$, either $c$ predicts perfectly ($\text{difference} = 0$) 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 \((x, y) \rightarrow (x, I(y > t), qI(y > t) + (1 - q)I(y < t))\)

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.
normalized performance

Quantile Prediction Performance

linear

quanting/log. reg.

quanting/J48

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
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

For A into a learner for B.

Reductions turn a (black-box) learner
Always applicable
Weaker (relative) statements
No assumptions

Agnostic Learning
Bayes Learning
VC Learning
PAC Learning

Theorem: consistency
- Probability: has small error with high
- to output a hypothesis $h$ in $H$
- most poly many i.i.d. examples
- The learner needs to observe at
- Concept class $C$, hypothesis class $H$
Multiclass Classification

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

- Multiclass Classifier: $c_m : X \rightarrow \{1, \ldots, k\} = \text{predictor}$

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

\[
e(D, c_m) = \Pr_{(x,y) \sim D} (c_m(x) \neq y)
\]
ECC Transformmaton

Binary examples

\[ ((\{1, 2, 3, 5\} \subseteq \{x\} \land \{1, 2\} \subseteq \{x\}) \]

Multiclass examples

\[ \{y \in \mathbb{N} \mid \{1, \ldots, n\} \subseteq \{y\} \land \{1, 2\} \subseteq \{y\} \} \]

"Is the label in 5 or not?"

defines a binary problem:

Each row (subset of labels 5)
Error Transformation

\((\mathcal{S} \ni f_i, I, \langle S, i \rangle) \) 

Draw \( \mathcal{D} \) \( \sim \) \( f_i, I \) 

draw a random row \( S \), output \( \mathcal{D} \) 

\[ \{ \{ 1 \} \times \overline{X} \}_{D \text{ over } \mathcal{D}} \]

Binary distribution

\[ \{ \{ 1, \ldots, \ell \} \times \overline{X} \}_{D \text{ over } \mathcal{D}} \]

Multiclass distribution

ECOC Transformation
classifier with respect to $D$.

Output the closest label

Decoding:

Bound the multiclass error

ECOC Prediction

(no ties broken randomly)

Predictions

Binary

1 2 ...
ECOC Analysis

Use one classifier trick $c(x, s) = c_s(x)$. Let $\text{ECOC}(D_m) =$ 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.
\[ (\eta, \theta) < 0 \]

fails to provide an optimal multiclass classifier:

\[ \min c = \arg \min c \in \text{ECOC}(D) \]

ECOC with \( c^* \). There exists \( D \) such that for all codes:

ECOC Inconsistency:

2) Inconsistency:

1) Not interesting if the error rate of the oracle is \( \frac{1}{4} \).

Two Little Problems
PECO: Probabilistic ECOC
Probabilistic Error Correcting Output Code

As ECOC, except use Probing to get probabilistic predictions.

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

<table>
<thead>
<tr>
<th>Binary Problem</th>
<th>Label</th>
<th>Predictions</th>
<th>Predictions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>0.91</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0.91</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0.55</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0.46</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0.45</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0.54</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0.54</td>
</tr>
</tbody>
</table>

| sum | 1.08 | 1.10 | 1.82 | 2.00 |
PECOC Analysis

Use one classifier trick $c(x, s, p) = c_{sp}(x)$. Let $\text{PECOC}(D_m) =$ 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.
If $b = \text{number of binary problems, distance between codewords } = \frac{b}{2}$

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 \( eb \) 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.

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

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

... and this turns out to be stable.

Theorem: (PECOC’ Translation) For all \( c : X \times [0, 1] \rightarrow \{0, 1\} \), 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: $c_m : X \rightarrow \{1, ..., k\} = \text{predictor}$

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

$$e_{cs}(D_{cs}, c_m) = E_{(x, \ell) \sim D_{cs}}[\ell_{cm}(x)]$$
**Sensitive Error Correcting Output Code**

\[ \text{SECOC} = \text{cost sensitive} \Rightarrow \text{importance weighted classification reduction, in two parts. Uses a code matrix } M: \]

<table>
<thead>
<tr>
<th>Label</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subset</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
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 $(x, \bar{\ell}) \in S$, let $|\bar{\ell}| = \sum_y \ell_y$ and $\ell_s = \sum_{y \in s} \ell_y$.

   (b) For random $t$ in $[0, 1]$:
      
      Let $c_{st} = A(\{(x, I(\ell_s \geq t|\bar{\ell}|), |\ell_s - |\bar{\ell}|t|) : (x, \bar{\ell}) \in S\})$.

2. return $\{\ell_{st}\}$
SECOC, the prediction algorithm

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

return \( \min_y E_s E_t [I(y \in s) c_{st}(x) + I(y \not\in s)(1 - c_{st}(x))] \)
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 \((x, \vec{\ell})\),

\[ e_{cs}(D_{cs}, \text{SECOC}_c) - \min_{c'} e_{cs}(D_{cs}, c') \leq 4 \sqrt{(e(\text{SECOC}(D_{cs}), c) - \min_{c'} e(\text{SECOC}(D_{cs}), c'))E_{x, \vec{\ell} \sim D_{cs}} |\vec{\ell}|} \]
Proof (sketch only, much like PECOC)

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

2. \( E_s \frac{E_{\ell \sim D}^*[\ell_s]}{E_{\ell \sim D}^*[|\ell|]} = \frac{E_{\ell \sim D}^*[\ell_y]}{E_{\ell \sim D}^*[|\ell|]} + \frac{1}{2} \) 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

\[
\Rightarrow E_{x \sim D_{cs}} \left( \text{Predict}(c, x, y) E_{\ell' \sim D|x} \left[ |\ell'| \right] - E_{\ell' \sim D|x} \left[ \ell'_y \right] \right)^2 \\
\leq 4 \left( \epsilon(\text{SECOC}(D_{cs}), c) - \min_{c'} \epsilon(\text{SECOC}(D_{cs}), c') \right) E_{(x, \ell) \sim D_{cs}} |\ell|
\]
Some Things to Think About

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

Which code do you use in practice?

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

Formal study of learning reductions is relatively new.

- The limits of the possible are not entirely known.

- 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.)
Importance Weighted Classification

Outlier Detection

Reinforcement Learning with Generative Model

Classification Problems

All Bounded loss Classification

Cost Sensitive Classification

Classification

Multiclass Classification

Ranking

Quantile Regression

Costing, etc...

Probing

Mean Regression

PECOC

$4r^{0.5}$

Squared error

$\sum_{i=1}^{n} c_i$

$\max c_i$

$\sum_{i=1}^{n} c_i$

$\sum_{i=1}^{n} c_i$

$4 (r E (max cost))^{0.5}$

Reinforcement Learning with Generative Model

Regression

SECOC

$4 (r E (sum of costs))^{0.5}$

Importance Weighted Classification
Related Reading


