Reductions

Problem 1: the problem you have to solve for a real application
Problem 2: a well-studied problem in machine learning
Problem instance: training data and (implicitly) a probability distribution $P$
Solution: prediction functions
$A$: the latest, greatest learning algorithm for Problem 2

Importance-weighted classification

Problem:
Setting: Random triple $(X, Y, C) \sim P$ for some probability distribution $P$ over $\mathcal{X} \times \mathcal{Y} \times \mathbb{R}_+$.
$C =$ importance weight (or “cost”) for labeled example $(X, Y)$.
Goal: Function $f: \mathcal{X} \rightarrow \mathcal{Y}$ with small importance-weighted error:
$\mathbb{E}[C \cdot 1\{f(X) \neq Y\}]$.

Problem instance:
Training data $S$: collection of triples $(x, y, c) \in \mathcal{X} \times \mathcal{Y} \times \mathbb{R}_+$, presumed to be drawn i.i.d. from $P$.

Where it comes up:
Class-specific weights: e.g., $C = 100 \Leftrightarrow Y = 0$ (and $C = 1$ otherwise).
Input-specific weights: e.g., $C = 100 \Leftrightarrow X \in \mathcal{X}_0$ (and $C = 1$ o.w.).
Boosting, domain adaptation, causal inference, ... (Note: many learning algorithms natively handle importance weights.)
Would like to reduce to (unweighted) classification.

Examples

0. Problem: binary classification
   Reduction: boosting
   (Reduces problem to binary classification.)
1. Problem: importance-weighted classification
   Reduction: rejection sampling
   (Reduces problem to unweighted classification.)
2. Problem: multi-class classification
   Reduction: One-Against-All
   (Reduces problem to binary classification.)
The rejection sampling reduction

**Main idea:** Transform training data $S$ so it looks like it came from a distribution $P'$, where

$$
\mathbb{E}_{(X,Y,C) \sim P} \left[ C \cdot 1\{f(X) \neq Y\} \right] \propto \mathbb{E}_{(X',Y') \sim P'} \left[ 1\{f(X') \neq Y'\} \right].
$$

**Instance mapping procedure**

**Input** Training data $S$ from $X \times Y \times R_+$. 

1. Initialize $S' = \emptyset$.
2. Let $c_{\text{max}} := \max_{(x,y,c) \in S} c$.
3. for each $(x, y, c) \in S$ do
4. Toss a coin with $\Pr(\text{heads}) = \frac{c}{c_{\text{max}}}$.
5. If heads, keep example—put $(x, y)$ into $S'$.
6. If tails, discard example.
7. end for
8. return Training data $S'$ from $X \times Y$.

**Solution mapping procedure:** identity map

Often useful to repeat this several times, and combine learned classifiers using majority vote.

**Why rejection sampling works:** (Assume for simplicity that $c_{\text{max}} = 1$.) Define random variable

$$
Q := 1\{\text{Keep example } (X,Y)\}
$$

which, after conditioning on $(X, Y, C)$, has mean $C$.

Distribution of examples in $S'$ is same as that of $(X, Y)$ conditioned on $Q = 1$.

Moreover,

$$
\mathbb{E} \left[ Q \cdot 1\{f(X) \neq Y\} \right] = \mathbb{E} \left[ \mathbb{E} \left[ Q \cdot 1\{f(X) \neq Y\} \mid (X, Y, C) \right] \right]
$$

$$
= \mathbb{E} \left[ C \cdot 1\{f(X) \neq Y\} \right]
$$

**Conclusion:**

Error rate w.r.t. $P'$ $\propto$ importance-weighted error rate w.r.t. $P$.

---

Multi-class classification

**Problem:**

- **Setting:** Random pair $(X, Y) \sim P$ for some probability distribution $P$ over $X \times \{1, 2, \ldots, K\}$.
- **Goal:** Function $f : X \to Y$ with small prediction error $P(f(X) \neq Y)$.

**Problem instance:**

- Training data $S$: collection of pairs $(x, y) \in X \times \{1, 2, \ldots, K\}$, presumed to be drawn i.i.d. from $P$.

Would like to reduce to binary classification.

---

One-Against-All reduction

**Main idea:** Create $K$ binary classification problems

given $x \in X$, predict whether or not $y = i$.

Create $K$ examples from each $(x, y) \in S$:

$$(x, y) \rightarrow \begin{cases}
(x, 1\{y = 1\}) & \rightarrow S'_1 \\
(x, 1\{y = 2\}) & \rightarrow S'_2 \\
\vdots & \vdots \\
(x, 1\{y = K\}) & \rightarrow S'_K
\end{cases}$$
One-Against-All reduction

Instance mapping procedure
Input Training data \( S \) from \( \mathcal{X} \times \{1, 2, \ldots, K\} \).
1: Initialize empty sets \( S'_1, S'_2, \ldots, S'_K \).
2: for each \((x, y) \in S\) do
3: for each \(i = 1, 2, \ldots, K\) do
4: Put \((x, \mathbb{1}\{y = i\}) \in \mathcal{X} \times \{0, 1\}\) into \(S'_i\).
5: end for
6: end for
7: return Training data sets \( S'_1, S'_2, \ldots, S'_K \) from \( \mathcal{X} \times \{0, 1\} \).

Solution mapping procedure
Input \( K \) binary predictors \( f'_1, f'_2, \ldots, f'_K : \mathcal{X} \to \{0, 1\} \).
return Function \( f : \mathcal{X} \to \{1, 2, \ldots, K\} \) where \( f(x) = \arg \max_{i \in \{1, 2, \ldots, K\}} f'_i(x) \) (breaking ties arbitrarily).

This should seem weird!

Problem with OAA

OAA multi-class predictor:
\[
f(x) = \arg \max_{i \in \{1, 2, \ldots, K\}} f_i'(x).
\]

Only get correct classification on \((x, y)\) if \(f'_y(x) = 1\) and \(f'_i(x) = 0\) for all \(i \neq y\).
(Could err if any of the \(f'_i\) errs!)

Solution: use conditional probability estimation
\[
f_i'(x) = \text{estimate of } P(Y = i | X = x).
\]

Comparing OAA and ECOC

Empirical comparison

Many reductions for multi-class—not all work equally well!

- Eight multi-class problems (from the UCI repository).
- \( A = \text{classregtree} \) from the MATLAB statistics toolbox, estimate conditional probabilities using square loss.
- Compare One-against-all (OAA) to Error Correcting Output Codes (ECOC).

Data set | Number of classes | OAA | ECOC |
--- | --- | --- | --- |
ecoli | 8 | 0.0985 | 0.0517 |
glass | 6 | 0.3874 | 0.3462 |
pendigits | 10 | 0.0985 | 0.0517 |
satimage | 6 | 0.1679 | 0.1376 |
soybean | 19 | 0.6580 | 0.5993 |
splice | 3 | 0.0642 | 0.0699 |
vowel | 11 | 0.6356 | 0.5780 |
yeast | 10 | 0.4893 | 0.4479 |

Summary

- Reductions: reuse existing technology to solve new problems.
  - Multi-class (OAA, ECOC, tournaments, \ldots)
  - Multi-label prediction
  - Ranking
  - Sequence prediction
  - \ldots

- Lots of different problems and objectives beyond binary classification and prediction error—can be application-/domain-specific.
1. Concept of reductions.
2. Reduction for importance-weighted classification.
3. OAA reduction for multi-class.
4. Importance of conditional probability estimation.