# Lecture 8: Classification with imbalanced classes and costs

Marina Meilă
mmp@stat.washington.edu

Department of Statistics
University of Washington

November, 2014

Training cost-sensitive classifiers

What is the goal? Performance criteria

# Cost-sensitive classifiers

- Sometimes, misclassification error $L_{01}$ loss is not appropriate for the problem
- Instead, asymmetric (or imbalanced) costs:
  - $c_+$ = loss (cost) of misclassifying true $+1$ as $-1$ (type II error)
  - $c_-$ = loss (cost) of misclassifying true $-1$ as $+1$ (type I error)
- How shall we train classifiers to minize asymmetric loss functions?
- Some classifiers can take the costs $c_\pm$ as input for training: they are called **cost-sensitive** classifiers[1]
  - Examples: CART, SVM, AdaBoost can be modified to "optimize" imbalanced costs (usually heuristic methods)
- Probabilistic classifiers (for which $L = -\ln P(Y|X)$) cannot be trained with costs (by definition). But cost can be incorporated after training, as a form of prior or bias (Examples: Logistic regression, generative classifiers)

[1] This actually means cost-sensitive trainable classifier.

# Cost-sensitive SVM

- Several proposals exist.
- The simplest one: $\min_{w,b,\xi} \frac{1}{2}||w||^2 + C(c_+ \sum_{y^i=+} \xi_i + c_- \sum_{y^i=-} \xi_i)$ s.t. *usual constraints*
- A risk minimization motivated framework [**?**]

$$
\begin{aligned}
\min_{w,b,\xi} \quad & \frac{1}{2}||w||^2 + C\left[c_+ \sum_{y^i=+} \xi_i + (2c_- - 1) \sum_{y^i=-} \xi_i\right] && (1)\\
\text{s.t.} \quad & w^T x^i + b \geq 1 - \xi_i \text{ for } \mathrm{y}^{\mathrm{i}} = +1, && (2)\\
& w^T x^i + b \leq \frac{1}{2c_- - 1} + \xi_i \text{ for } \mathrm{y}^{\mathrm{i}} = -1 && (3)
\end{aligned}
$$

assuming $c_+ \geq 2c_- - 1 \geq 1$.

# Cost-sensitive AdaBoost

- Several proposals exist, most of them heuristic.
- A risk minimization motivated framework [?]

  Use the exponential loss function

  $$\phi_{c_+,c_-}(y,f) = \begin{cases} \exp(-c_+ f), & y = +1 \\ \exp(c_- f), & y = -1 \end{cases} \tag{4}$$

  instead of the cost-insensitive exponential loss $\phi(yf) = \exp(-yf)$.

  Note that this implies each weak learner is trained in a cost-insensitive manner, but with weights that emphasize the positive examples.

# Imbalanced classes

**Problem** Binary classification

- In many real-world problems, one class is much more frequent than the other(s). Usually, the rare class is what makes the problem interesting in first place.

Examples Testing for most diseases, fraud detection, finding splice junctions in the DNA, face detection in an image, predicting user behavior on the web (most visits to e.g Amazon don't end with a purchase).

- Why the special attention?

A1 Let $p = P[Y = 1] \ll 1$ be the (true) probability of the rare class. (Running example, assume $p = 0.01$ and the sample size $N = 10,000$).

Then, the trivial classifier $f(x) = -1$ has $1 - p = 99\%$ accuracy!

- So, if it's not important to detect the rare class 1 examples, stop reading.

  In other words: train a classifier only when it's important to separate the class 1 data from the others

# Alternative performance measures

- **True positives rate** $p_{TP} = \frac{\#y=+,\hat{y}=+}{\#y=+}$
- **False negative rate** $p_{FN} = \frac{\#y=+,\hat{y}=-}{\#y=+}$

  $p_{TP} + p_{FN} = 1$
- **True negative rate** $p_{TN} = \frac{\#y=-,\hat{y}=-}{\#y=-}$
- **False positive rate** $p_{FP} = \frac{\#y=-,\hat{y}=+}{\#y=-}$

  $p_{TN} + p_{FP} = 1$

- **Precision** $Prec = \frac{\#y=+,\hat{y}=+}{\#\hat{y}=+}$
- **Recall** $Rec = p_{TP}$
- $F_\beta = \frac{(1+\beta^2)PrecRec}{\beta^2 Prec+Rec}$ weigths precision and recall by a coefficient $\beta$

  Note that $\frac{1}{F_{\beta=1}} = \frac{1}{2}\left(\frac{1}{Prec} + \frac{1}{Rec}\right)$ (harmonic mean)
- $F1 = \sqrt{PrecRec}$.
- . . .
- **ROC (Receiver Operating Characteristic) curve** (later in this lecture)

# Imbalanced costs and imbalanced classes

A2 Let $c_+, c_-$ be the losses (costs) of misclassifying true $+1, -1$ examples.

- **Decision theoretic point of view** Take decision that minimizes expected cost for input $x$

$$\hat{y} = +1 \quad \text{iff} \; \underbrace{P[Y=-1|x]c_-}_{E[\text{cost of } y=-1]} < \underbrace{P[Y=+1|x)c_+}_{E[\text{cost of } y=+1]} \tag{5}$$

- When the output of a clasifier represents (an estimator of) $P[Y=1|x]$ (e.g for logistic regression), then (5) gives the optimal decision rule.
- In particular, for a generative classifier, that estimates $g_\pm(x) \equiv P[X|Y=\pm 1]$, the optimal decision rule is a likelihood ratio

$$\hat{y} = 1 \quad \text{iff} \quad \frac{g_+(x)}{g_-(x)} \geq \frac{(1-p)c_-}{pc_+} = \tau_{generative} \tag{6}$$

- For discriminative classifiers (previous section)
  - interpret $f(x)$ as uncalibrated $P[Y=+|x]$ or
  - use **cost-sensitive** versions

# What if $c_\pm$ are not known at the time of training?

- $c_\pm$ may not be known, or may be choices of convenience (e.g. the cost of not detecting a serious conditions is variable)
- $p$ may also be imprecisely known
  - because the sample distribution of classes is not the same as the population distribution (e.g in clinical trials)
  - because the classifier will be deployed on different populations, each with its own class distribution
  - because for very small $p$, estimating it from a sample has large relative error

Let $p = 0.01,\ N = 10,000$. Then

- $E[\hat{p}] = p,\ Var(\hat{p}) = p(1-p)/N \approx p/N$
- Relative error $(\hat{p}) = \frac{\sqrt{Var(\hat{p})}}{p} \approx \frac{1}{\sqrt{pN}} = 0.1$

Then

1. the decision rule (5) will depend on a **threshold** parameter $\tau$ to be estimated (or re-estimated) after the classifier is trained

$$\hat{y} = +1 \quad \text{iff} \quad P[y = +|x] \geq \tau \tag{7}$$

2. estimating the performance should be done in a way that is independent of the threshold $\tau$

# The ROC Curve

- Intuitition: if $\tau > \tau'$, then $\hat{y}(\tau) \leq \hat{y}(\tau')$. In words, if the threshold is increased, some positive $\hat{y}$ will become negative, and all negative $\hat{y}$ will stay so. Hence $p_{TP}, p_{FP}$ are both non-increasing with $\tau$.
- The **ROC curve**[2] plots $p_{TP}(\tau)$ vs. $p_{FP}(\tau)$ for all $\tau$ values between $(-\infty, \infty)$ [that make sense]
- Extremes
  - for $\tau$ very large, $p_{TP} = p_{FP} = 0$
  - for $\tau$ very negative, $p_{TP} = p_{FP} = 1$
- Ideally: there is a $\tau$ for which $p_{TP} = 1$, $p_{FP} = 0$. Then, the whole ROC curve is on the boundary of the square $[0, 1] \times [0, 1]$.

- **AUC** denotes the **A**rea **u**nder the ROC **C**urve.
  - Ideally: $AUC = 1$. In all other cases $AUC < 1$
  - $AUC \approx 1$ is great
  - For random guessing, $p_{TP}(\tau) \approx p_{FP}(\tau)$, hence $AUC \approx 0.5$.
  - Therefore $AUC \approx 0.5$ is very bad.
- Classifiers are compared by their AUC on test set

Figure: example of ROC curve

[2] A term from signal processing.

# Practical construction of an ROC curve

1. With **cost-sensitive** classifiers
   - Try different cost ratios $\frac{c_+}{c_-}$. (E.g, fix $c_- = 1$, change $c_+$)
   - For each, train a cost-sensitive classifier, calculate its $p_{TP}$, $p_{FP}$ on a test set. This gives a point on the ROC curve.
   - Add the points $(0, 0)$, $(1, 1)$. Connect the dots to obtain the ROC curve.

2. With real-valued (cost-insensitive) classifiers

   - Some classifiers output (an estimate of) $P[Y = +|x]$ (these are called calibrated). E.g logistic regression, generative classifiers, logistic output neural networks (trained with log-likelihood)
   - Others output a real valued function $f(x)$. Assumption: $f(x) \nearrow$ implies $P[Y = +|x] \nearrow$ (these are called uncalibrated). Examples: SVM, discriminative linear and quadratic classifiers, AdaBoost, Random Forests, any other ensemble classifiers (bagged, stacked)
   - Some are $\pm 1$-valued classifiers. Examples: $K$-nearest neighbors, CART.
     - First option: turn them into "probabilistic" classifiers. E.g for K-NN, ouput ratio of majority class over $K$; for CART, do the same in the current leaf.
     - Second option: **Meta-learning** (e.g MetaCost algorithm). Use a form of bagging (or bootstrap) to estimate $P[Y = +|x^i]$.

   - In either case, given a test set of size $N'$, sorting $i = 1, \ldots N'$ by $f(x^i)$ (in descending order) lists the test set data in decreasing order of (our estimate of) $P[y^i = +|x^i]$.
   - No matter what $p, c_\pm$ are, applying (5) can produce $N' + 1$ distinct classifiers, $j = 0, \ldots N'$.

     Classifier $j$ sets $\hat{y}^{[1]:[j]} = +1$, and the rest $-1$.
   - This gives us $N' + 1$ points on the ROC curve, including the ends.