## 8.2.2 Maximum likelihood inference

Today
- Boosting (AdaBoost : the first popular boosting algorithm)
- statistical interpretation of boosting (L2 Boosting)
- likelihood-based boosting (together with model-based boosting (gradient boosting), statistical boosting)

$Y_i$ iid with density $p(y_i;\theta)$

e.g. $Y_i \sim N(\mu,\sigma^2) \rightarrow p(y_i;\underbrace{\mu,\sigma^2}_{\theta}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(y_i-\mu)^2\right\}$

$L(\theta;y) = \prod_{i=1}^{N} p(y_i;\theta)$ likelihood

$\ell(\theta;y) = \sum_{i=1}^{N} \log p(y_i;\theta)$ log-likelihood

$\hat{\theta} = \arg\max_\theta \ell(\theta;y) = \arg\max_\theta L(\theta;y)$

$\ell_\theta(\theta;y) = \frac{\partial \ell(\theta;y)}{\partial\theta}$ score function

$\ell_{\theta\theta}(\theta;y) = \frac{\partial^2 \ell(\theta;y)}{\partial\theta\,\partial\theta^T}$

$\rightarrow j(\theta) = -\ell_{\theta\theta}(\theta;y)$ observed information

$\rightarrow i(\theta) = E_\theta[j(\theta)]$ expected information

If $\theta_0$ is the true parameter

$\hat{\theta} \xrightarrow{n\to\infty} N(\theta_0; j(\theta_0)^{-1})$

Estimation

$\hat{\theta} \sim N(\hat{\theta}; j(\hat{\theta})^{-1})$ or $\hat{\theta} \sim N(\hat{\theta}; i(\theta)^{-1})$

confidence intervals $\hat{\theta} \pm z_{1-\frac{\alpha}{2}} \sqrt{j^{-1}(\hat{\theta})}$

# 10 Boosting

Leo Breiman: "[boosting is] the best off-the-shelf classifier in the world"

- originally developed for classification;
- translated into the statistical world and use for all purposes (regression, ...)
  - extended in its use;
  - interpretable (GAM)

Starting challenge:
"can a committee of blackheads somehow arrive at a highly reasoned decision, despite the weak judgement of the individual members?"

goal: . obtain a good classifier
. apply "weak estimators", in the context of classification classifiers which lead to a solution only slightly better than a random choice

idea: repeatedly (iteratively) apply a weak estimator to modifications of the data

AdaBoost

at each iteration AdaBoost gives more weight to the missclassified observations

Consider a two class classification problem

$Y_i \in \{-1, 1\}$

$X_i$ the vector of inputs

AdaBoost algorithm

① inizialize, weights are $\left(\frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}\right) = w^{[0]}$

② for m from 1 to m_stop (M)

a) fit the weak estimator $\hat{g}(x)$ to the weighted data;

b) compute the weighted in-sample missclassification rate

$$err^{[m]} = \frac{\sum_{i=1}^{N} w_i^{[m-1]} \mathbb{1}[y_i \neq \hat{g}^{[m]}(x_i)]}{\sum_{i=1}^{N} w_i^{[m-1]}}$$

c) compute $\alpha_m = \log\left(\frac{1 - err^{[m]}}{err^{[m]}}\right)$

$\alpha_m$ is used to weight the contribution of the $\hat{g}^{[m]}(x)$ to the final estimate (classification)

d) update the weights

$$\tilde{w}_i = w^{[m-1]} \exp\left\{\alpha_m \underbrace{\mathbb{1}[y_i \neq \hat{g}^{[m]}(x_i)]}_{\text{reweight only missclassified observations}}\right\} \quad i = 1, \dots, N$$

OR

$$w_i^{[m]} = \frac{\tilde{w}_i}{\sum_{i=1}^{N} \tilde{w}_i}$$

③ compute final result

$$\hat{f}_{AdaBoost} = \mathrm{sgn}\left(\sum_{m=1}^{m\_stop} \alpha^{[m]} \hat{g}^{[m]}(x)\right)$$

Example

step [0] $w = \left(\frac{1}{10}, \frac{1}{10}, \dots, \frac{1}{10}\right)$

step [1] $err = \dfrac{\sum_{i=1}^{N} \frac{1}{10} \mathbb{1}[y_i \neq \hat{g}_i]}{\sum_{i=1} \frac{1}{10}} = \dfrac{\frac{3}{10}}{\frac{10}{10}} = 0.3$

$\alpha_1 = \log \frac{1-err}{err} = \log 0.7 - \log 0.3 \simeq 0.84$

$\tilde{w} = \Big(\underbrace{\exp(0.84)\frac{1}{10}}_{\text{missclassifier in the first iteration}}\overset{\approx 0.23}{,}\ 0.23, 0.23, 0.1, \dots, 0.1\Big)$

$w^{[1]} \approx (0.17, 0.17, 0.17, 0.07, \dots, 0.07)$

---

Statistical view of Boosting
- functional gradient descendent algorithm
- forward stagewise (additive) modelling

→ see AdaBoost as an iterative procedure to minimize a loss-function, in particular an exponential loss-function

$$L(y, f(x)) = \exp\{-y f(x)\}$$

Consider a generic step $m$

- the current classifier is $\hat{f}^{[m-1]} = \sum_{k=1}^{m-1} \alpha_k \hat{g}^{[k]}(x)$ $\underline{1 \dots m-1 \text{ given } m \text{ looking}}$
  (if the algorithm had stopped at the m-1 iteration)
- the goal is to find $(\alpha_m, g_m) = \arg\min_{\alpha_m g_m} \sum_{i=1}^{N} \exp\{-y \underbrace{\sum_{k=1}^{m} \alpha_k g^{[k]}}_{f}\}$

$$\alpha_m, g_m = \arg\min_{\alpha, g} \sum_{i=1}^{N} \exp\{-y(\underbrace{\sum_{k=1}^{m-1} \alpha_k \hat{g}^{[k]}}_{\hat{f}^{[m-1]}} + \alpha g)\}$$

$$= \arg\min_{\alpha, g} \sum_{i=1}^{N} w_i^{[m-1]} \exp\{-y\alpha g\} \qquad (*)$$

where $w_i = \exp\{-y \hat{f}^{[m-1]}\}$, which do not depend neither on $\alpha$ nor on $g$ (given from the previous iteration)

$y \in \{-1, 1\}$
$g \in \{-1, 1\}$

- two step procedure: first we minimize with respect to $g$

$\arg\min_g \sum_{i=1}^{N} w_i^{[m-1]} \exp(-y\alpha g)$ if $\boxed{y=g} \to yg = \boxed{1}$; if $y \neq g \to yg = -1$

$= \arg\min_g \left\{\sum_{g=y} w_i^{[m-1]} e^{-\alpha} + \sum_{g \neq y} w_i^{[m-1]} e^{\alpha}\right\}$

$\pm e^{-\alpha} \sum_{g \neq y} w_i$

$= \arg\min_g \left\{e^{-\alpha} \sum_{i=1}^{N} w_i^{[m-1]} + (e^{\alpha} - e^{-\alpha}) \sum_{g \neq y} w_i\right\}$

$\arg\min \left\{\sum_{g=y} w_i^{[m-1]} e^{-\alpha} + \sum_{g \neq y} w_i^{[m-1]} e^{-\alpha} - \sum_{g \neq y} w_i^{[m-1]} e^{-\alpha} + \sum_{g \neq y} w_i^{[m-1]} e^{\alpha}\right\}$

$= \arg\min_g \left\{e^{-\alpha} \sum_{i=1}^{N} w_i^{[m-1]} + (e^{\alpha} - e^{-\alpha}) \sum_{i=1}^{N} w_i^{[m-1]} \mathbb{1}[g_i \neq y_i]\right\}$

$g = \arg\min_g \left\{\sum_{i=1}^{N} w_i^{[m-1]} \mathbb{1}[g_i \neq y_i]\right\}$

$$= \arg\min_{\alpha, g} \sum_{i=1}^{n} w_i^{[m-1]} \exp\{-y\alpha g\}$$

$$\arg\min_{\alpha} \sum w_i^{[m-1]} \exp\{-y\alpha g\} \qquad \frac{\partial L}{\partial \alpha} = \sum$$

$$y = g \longrightarrow \sum_{g=y} w_i^{[m-1]} \exp\{-\alpha\}$$

$$y \neq g \longrightarrow \sum_{g \neq y} w_i^{[m-1]} \exp\{\alpha\}$$

$$\frac{\partial L}{\partial \alpha} = -\sum_{g=y} w_i^{[m-1]} \exp\{-\alpha\} + \sum_{g\neq y} w_i^{[m-1]} \exp\{\alpha\} = 0$$

multiply both terms for $e^{\alpha}$

$$= -\sum_{g=y} w_i^{[m-1]} + \sum_{g \neq y} w_i^{[m-1]} \exp\{2\alpha\} = 0$$

$$e^{2\alpha} \sum_{g\neq y} w_i^{[m-1]} = \sum_{g=y} w_i^{[m-1]}$$

$$e^{2\alpha} = \frac{\sum_{i=1}^{n} w_i^{[m-1]} \cancel{\sum_{i=1}^{n} w_i^{[m-1]}} - \sum_{g\neq y} w_i^{[m-1]} \cancel{\sum_{i=1}^{n} w_i^{[m-1]}}}{\sum_{g \neq y} w_i^{[m-1]} \cancel{\sum_{i=1}^{n} w_i^{[m-1]}}}$$

$$\alpha = \frac{1}{2} \log\left(\frac{1 - err}{err}\right)$$

where $err = \dfrac{\sum_{i=1}^{n} w_i^{[m-1]} \mathbb{1}(y \neq g)}{\sum_{i=1}^{n} w_i^{[m-1]}}$

- $\hat{g}^{[m]}$ minimizer of the weighted missclassification
- $\alpha = \frac{1}{2} \log\left(\frac{1-err}{err}\right)$

Our general classifier is updated as

$$\hat{f}^{[m]} = \hat{f}^{[m-1]} + \underbrace{\alpha_m}\,\underbrace{\hat{g}^{[m]}}$$

which causes the weights of the next iteration to be

$$w_i^{[m]} = w_i^{[m-1]} \cdot \exp\{\underbrace{-\alpha\, y_i \hat{g}_i^{[m]}}\}$$

Since $-y_i\hat{g}_i^{[m]} = -\sum_{g_i=y_i} 1 + \sum_{g_i\neq y_i}(+1) \pm \sum_{g_i\neq y_i} 1$

$$= 2\sum_{g_i\neq y_i} 1 - 1$$

$$w_i^{[m]} = w_i^{[m-1]} \cdot \exp\{+\alpha\,(2\sum_{g_i\neq y_i} 1 - 1)\}$$

$$= w_i^{[m-1]} \exp\{2\alpha\, \mathbb{1}(y_i\neq g_i) - \alpha\}$$

$$= w_i^{[m-1]}\, e^{-\alpha}\, e^{2\alpha\, \mathbb{1}(y_i\neq g_i)}$$

$e^{-\alpha}$: constant for each observation → can be ignored

$2\beta = 2\alpha$ is $\alpha$ in the algorithm

→ AdaBoost minimizes the exponential loss criterion by a forward stagewise procedure

Note:
- this statistical view allows us to interpret the results of the procedure.
  In particular, it can be showed that the minimizer of the exponential loss

$$f^*(x) = \arg\min_{f(x)} E_{Y|x}[e^{-yf(x)}] = \frac{1}{2}\log\frac{Pr[Y=1|x]}{Pr[Y=-1|x]}$$

alternatively: $Pr[Y=1|x] = \dfrac{1}{1+e^{-2f^*(x)}}$

$\frac{1}{2}$ the log-odds for $Pr[Y=1|x]$ → Ex 10.2

- other loss-functions lead to the same minimizer, for example the negative log-likelihood

$$\tilde{\pi} = Pr[Y=1|x] = \frac{e^{f(x)}}{e^{f(x)}+e^{-f(x)}} = \frac{1}{1+e^{-2f(x)}}$$

- $y' = \frac{y+1}{2} \in \{0;1\}$

then

$$\ell(\tilde{\pi}) = y'\log\tilde{\pi} + (1-y')\log(1-\tilde{\pi})$$

$$\Rightarrow -\ell(\tilde{\pi}) = \log_2(1+e^{-2yf(x)})$$