Lecture November 6

Boosting

- algorithm based on a weak learner/base learner
  apply weak learner sequentially to weighted version of the data. More weight to misclassified data from previous rounds

$$\min_f C(f) = \min_f \sum_{i=0}^{n-1} \mathcal{L}(y_i, f(x_i))$$

$$f(x_i) = \boxed{\sum_{m=0}^{M} \underline{\beta_m} \, b(x_i; \gamma_m)}$$

$$= f_0(x_i) + f_1(x_i) + \dots f_M(x_i)$$

Typical Loss functions

- squared error:

$$\frac{1}{2}(y_i - f(x_i))^2$$

$$f(x_i) = \underline{f_{m-1}(x_i)} + \beta_m b(x_i, \gamma_m)$$

$$\frac{1}{2}\left(r_{im} - \beta_m b(x_i; \gamma_m)\right)^2$$

$$r_{im} = y_i - f_{m-1}(x_i)$$

- absolute error $|y_i - f(x_i)|$

with derivative $\operatorname{sign}(y_i - f(x_i))$

Classification

Exponential loss tailored to binary case $y_i \in \{-1, 1\}$

$$\mathcal{L}(y_i, f(x_i)) = e^{-y_i f(x_i)}$$

$$f(x_i) = \operatorname{sign}\left(\sum_{m=0}^{M} \beta_m b(x_i; \gamma_m)\right)$$

$$f(x_i) = \{-1, +1\}$$

Derivative $-y_i \exp(-y_i f(x_i))$

Puts emphasis on misclassified data. Tends to give too much weight to outliers. => Adaboost

- Logit loss : $\log(1 + e^{-y_i f(x_i)})$

The basic approach :

at iteration m we optimize :

$$(\hat{\beta}_m, \hat{\gamma}_m) = \arg\min_{\beta\, \gamma} \sum_{i=0}^{n-1}$$

$$\hookrightarrow \mathcal{L}(y_i, f_{m-1}(x_i) + \beta\, b(x_i; \gamma))$$

weak learner

Adaboost algo : Additive modeling.

data set $\{(x_0, y_0), (x_1, y_1) \ldots (x_{n-1}, y_{n-1})\}$

$f(x_i) \in \{-1, 1\}$

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

if $f(x_i)$ is correct

$$y_i \cdot f(x_i) = +1$$

else

$$y_i f(x_i) = -1$$

$$C(f) = \sum_{i=0}^{n-1} \exp(-y_i(f_{m-1}(x_i) + \beta b(x_i))$$

weak classifier

$$= \sum_{i=0}^{n-1} w_{im} \exp(-\beta y_i b(x_i))$$

$$w_{im} = \exp(-y_i f_{m-1}(x_i))$$

$f(x_i) \propto b(x_i)$

$$e^{-\beta \underbrace{y_i \cdot b(x_i)}_{1}} = e^{-\beta} \text{ correct}$$

if wrong

$$e^{+\beta}$$

$$C(\beta) = \boxed{e^{-\beta}\sum_{y_i = b(x_i)} w_i^m + e^{\beta}\sum_{y_i \neq b(x_i)} w_i^m}$$

$$e^{-\beta}\sum_{y_i \neq b(x_i)} w_i^m$$

$$+$$

$$= e^{-\beta}\sum_{i=0}^{n-1} w_i^m +$$

$$\left(e^{\beta} - e^{-\beta}\right)\sum_{i=0}^{n-1} w_i^m \, \mathbb{I}(y_i \neq b(x_i))$$

accuracy of misclassification

weighting misclassification

$$\frac{dC(\beta)}{d\beta} = \frac{d}{d\beta}\left[e^{-\beta}\sum_{y_i = b(x_i)} w_i^m + e^{\beta}\sum_{y_i \neq b(x_i)} w_i^m \right]$$

$$- e^{-\beta}\sum_{y_i = b(x_i)} w_i^m$$

$$+ e^{\beta}\sum_{y_i \neq b(x_i)} w_i^m = 0$$

$$-$$

$$e^{-\beta} \sum_{y_i = f(x_i)} w_{im} = e^{\beta} \sum_{y_i \neq f(x_i)} w_{im}$$

$$-\beta + \ln\left(\sum_{y_i = f(x_i)} w_{im}\right)$$

$$= \beta + \ln\left(\sum_{y_i \neq f(x_i)} w_{im}\right)$$

$$\beta = \frac{1}{2} \ln \left[\frac{\sum_{y_i = f(x_i)} w_{im}}{\sum_{y_i \neq f(x_i)} w_{im}} \right]$$

Define weighted error

$$\epsilon_m = \sum_{y_i \neq f(x_i)} w_{im} \Big/ \sum_{y_i = f(x_i)} w_{im}$$

$$\beta = \beta_m = \frac{1}{2} \ln\left(\frac{1 - \epsilon_m}{\epsilon_m}\right)$$

$$\epsilon_m = \frac{\sum_{i=0}^{n-1} w_{im} \, \mathbb{I}(y_i \neq f(x_i))}{\sum_{i=0}^{n-1} w_{im}}$$

The overall update

$$f_m(x) = f_{m-1}(x) + \beta_m b_m(x_i)$$

$$w_{i\,m+1} = w_{im}\, e^{-\beta_m y_i b_m(x_i)}$$

$$= w_{im}\, e^{\beta_m (2\mathbb{1}(y_i \neq b_m(x_i)) - 1)}$$

$$= w_{im}\, e^{2\beta_m \mathbb{I}(y_i \neq b_m(x_i))} \times e^{-\beta_m}$$

used $-y_i b_m(x_i) = -1$ if $y_i = b_m(x_i)$

Algorithm for Adaboosting

$w_i = \frac{1}{n}$ initial $f_0(x)$

for $m: 1$ to M DO

- fit a classifier $b_m(x)$ to the training set using weights w
- compute

$$\epsilon_m = \frac{\sum_{i=0}^{n-1} w_{im}\, \mathbb{I}(y_i \neq b_m(x_i))}{\sum_{i=0}^{n-1} w_{im}}$$

- compute

$$\beta_m = \frac{1}{2} \ln\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$$

– set new weights

$$w_{i\,m+1} = w_{im} \exp\left(\beta_m \cdot \mathbb{I}(y_i \neq b_m(x_i))\right)$$

$$f_m(x_i) = f_{m-1}(x_i) + \beta_m b_m(x_i)$$

End for

Return $f(x) = \operatorname{sign}\left(\sum_{m=0}^{M} \beta_m \times b_m(x)\right)$

Exponential Loss can be replaced by Logit Boost.

Till now we have derived new versions of boosting.

$$\hat{f} = \arg\min_{f} C(f)$$

where $f = (f(x_0), f(x_1), \ldots f(x_{n-1}))$ are the parameters, optimize f at a fixed set of points.

$$\boxed{f_m = f_{m-1} - \rho_m \cdot g_m}$$

$\left[\partial L(y_i, f(x_i)) \right]$

$$g_{im} = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right]_{f=f_{m-1}(x_i)}$$

$$\rho_m = \arg\min_{\rho} L(f_{m-1} - \rho g_m)$$

Can modify the algorithm by fitting a weak learner to approximate the negative gradient $\Rightarrow$ gradient Boosting

$$\boxed{\gamma_m = \arg\min_{\gamma} \sum_{i=0}^{n-1} \left(-g_{im} - b(x_i, \gamma)\right)^2}$$

Algorithm for Gradient Boosting

initialize $f_0(x)$

for $m = 1 : M$ DO

compute the gradient of residual

$$r_{im} = y_i - f_{m-1}(x_i)$$

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{[illegible]} \right]$$

$L \quad \partial f(x_i) \quad \Big|_{f(x_i) = f_{m-1}(x_i)}$

use weak learner to compute γ_m which minimizes

$$\sum_{i=0}^{n-1} \left(r_{im} - b(x_i;\gamma_m) \right)^2$$

update

$$f_m(x) = f_{m-1}(x) + b(x;\gamma_m)$$

end for

return $f_M(x) = f(x)$

Reminder on defs

TP = True positive (hit)

TN = TRUE negative
Correct rejection

FP = False positive
false alarm

FN = False negative,
miss

P = # real positive in data set

N = # negative cases

in data

TRUE positive rate

$$TPR = \frac{TP}{P} = \frac{TP}{TP+FN}$$

TNR = TRUE negative rate

$$= \frac{TN}{N} = \frac{TN}{TN+FP}$$

FALSE NEGATIVE RATE

$$FNR = \frac{FN}{FP+TN}$$

FALSE POSITIVE RATE

$$FPR = \frac{FP}{FP+TN}$$

Accuracy:

$$Acc = \frac{TP+TN}{P+N} =$$

$$\frac{TP+TN}{TP+TN+FP+FN}$$

F1 SCORE

$F1 \quad 2TP$

$$F1 = \frac{[illegible]}{2TP + FP + FN}$$