STAT406- Methods of Statistical Learning Lecture 18

Matias Salibian-Barrera

UBC - Sep / Dec 2017

Ensembles

- Ensembles of classifiers
- Combine classifiers trained on the same (or similar [e.g. bootstrapped]) data
- Consensus is reached by (equally weighted) voting or averaged estimated probabilities.
- Bagging and Random Forests are examples of ensembles.

Boosting

- Originally proposed for classification
- Main idea: sequentially re-train a simple classifier assigning more importance to points that were previously misclassified

Boosting

- The end result is a weighted average of all the classifiers
- Interesting ideas:
  - Not all components of the ensemble are treated equally
  - Members of the ensemble use information about other members
  - The underlying loss function has a “margin” (unlike 0-1 losses)

Boosting - AdaBoost.M1

Algorithm. Data $(y_{\mathbf{i}}, \mathbf{x}_{\mathbf{i}})$, with $y_{\mathbf{i}} \in \{-1, 1\}$

- Set initial weights $w_{\mathbf{i}} = 1/n$, $1 \leq \mathbf{i} \leq n$
- For $\mathbf{j} = 1, \ldots, K$
- Build a classifier $T_{\mathbf{j}}(\mathbf{x})$ to the data using weights $w_{\mathbf{i}}$, $1 \leq \mathbf{i} \leq n$

Boosting - AdaBoost.M1

- Let

$$e_{\mathbf{j}} = \sum_{\mathbf{i}=1}^{n} w_{\mathbf{i}} \, I\left(y_{\mathbf{i}} \neq T_{\mathbf{j}}(\mathbf{x}_{\mathbf{i}})\right) \Big/ \sum_{\ell=1}^{n} w_{\ell}$$

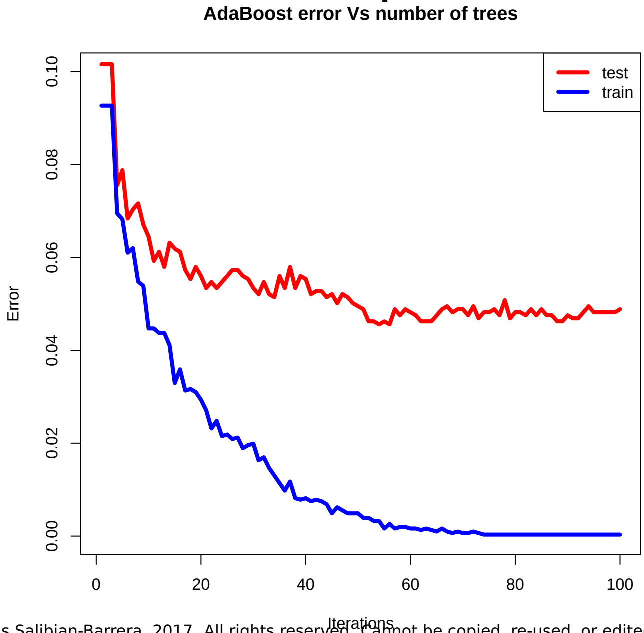
- Let $\alpha_{\mathbf{j}} = \log\left((1 - e_{\mathbf{j}})/e_{\mathbf{j}}\right)$ and

$$w_{\mathbf{i}} = w_{\mathbf{i}} \exp\left(\alpha_{\mathbf{j}} \, I\left(y_{\mathbf{i}} \neq T_{\mathbf{j}}(\mathbf{x}_{\mathbf{i}})\right)\right), \; \mathbf{i} = 1, \ldots, n$$

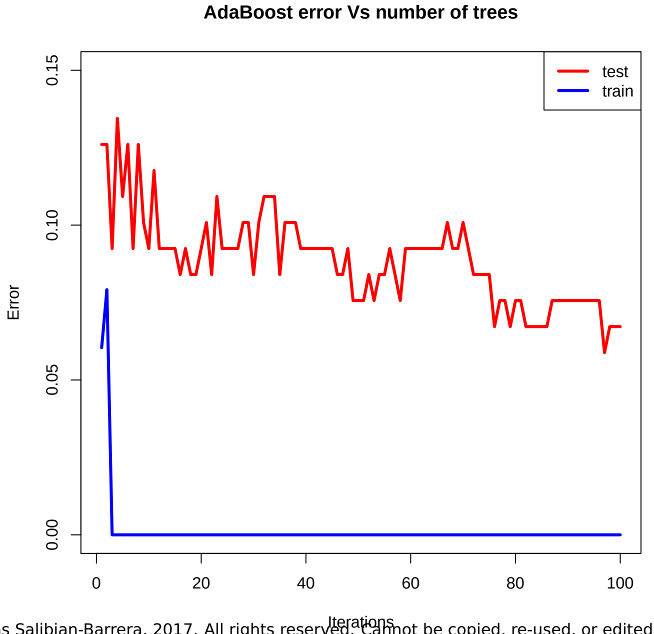
- Final classifier:

$$T(\mathbf{x}) = \operatorname{sign}\left(\sum_{\mathbf{j}=1}^{K} \alpha_{\mathbf{j}} \, T_{\mathbf{j}}(\mathbf{x})\right)$$

Error evolution - Spam data



Error evolution - Isolet



Boosting

- Boosting is fitting an **additive model**
- ... using a **forward search algorithm**
- ... and a **specific loss function**

Boosting

- Think of classifiers of the form

$$G(x) \;=\; \sum_{j=1}^{K} \beta_j \, f(\mathbf{x}, \gamma_j)$$

where $f(\mathbf{x}, \gamma_j)$ are simple base classifiers (e.g. trees)

Boosting

- Given a data set $(y_i, \mathbf{x}_i)$, $i = 1, \ldots, n$

$$\min_G \sum_{i=1}^{n} L\left(y_i, G(\mathbf{x}_i)\right) \; = $$

$$= \; \min_{\boldsymbol{\beta}, \gamma} \sum_{i=1}^{n} L\Big(y_i, \sum_{j=1}^{K} \beta_j \, f(\mathbf{x}_i, \gamma_j)\Big)$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_K)'$ and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_K)'$

Boosting

- Find approximate solutions sequentially

- Start with $f_0(\mathbf{x}) = 0$
- `for(j in 1:K)`
- Find

$$(\beta_j, \gamma_j) \ = \ \arg\min_{\beta,\gamma} \ \sum_{i=1}^{n} L\big(y_i, f_{j-1}(\mathbf{x}_i) + \beta\, f(\mathbf{x}_i, \gamma)\big)$$

- Let $f_j(\mathbf{x}) = f_{j-1}(\mathbf{x}) + \beta_j\, f(\mathbf{x}, \gamma_j)$

Counterexample

```
> set.seed(123)
> n <- 100
> x1 <- rnorm(n)
> x2 <- rnorm(n)
> y <- 2 - x1 + 6*x2 + rnorm(n, sd=.7)
> m1 <- lm(y~x1+x2)
> ( obj1 <- sum( resid(m1)^2 ) )
[1] 43.01311
>
> r1 <- y - mean(y)
> m2 <- lm(r1~x1-1)
> r2 <- resid(m2)
> m3 <- lm(r2~x2-1)
> ( obj2 <- sum( resid(m3)^2 ) )
[1] 109.3264
```


Boosting

- AdaBoost uses the following loss function

$$L\left(y, G(\mathbf{x})\right) \;=\; \exp\left(-y\, G\left(\mathbf{x}\right)\right)$$

- At the j-th iteration we have

$$\arg\min_{\beta,\gamma} \sum_{i=1}^{n} \exp\Big(-y_i \left(f_{j-1}(\mathbf{x}_i) + \beta f(\mathbf{x}_i, \gamma)\right)\Big)$$

$$\arg\min_{\beta,\gamma} \sum_{i=1}^{n} w_i^{(l-1)} \exp\Big(-\beta\, y_i f(\mathbf{x}_i, \gamma)\Big)$$

Boosting

- For any $\beta > 0$ the solution is the classifier $f(\mathbf{x}, \gamma)$ that minimizes

$$\sum_{y_i \neq f(\mathbf{x}_i, \gamma)} w_i^{(j-1)} = \sum_{i=1}^{n} w_i^{(j-1)} I\left(y_i \neq f(\mathbf{x}_i, \gamma)\right)$$

  which is a weighted missclassification error

Boosting

- Similarly we obtain

$$\beta_j \; = \; \frac{1}{2} \log \left(\frac{1 - e_j}{e_j} \right)$$

where

$$e_j \; = \; \sum_{i=1}^{n} w_i^{(j-1)} \, I \left(y_i \neq f(\mathbf{x}_i, \gamma_j) \right) \Big/ \sum_{i=1}^{n} w_i^{(j-1)}$$

Boosting

- We then update

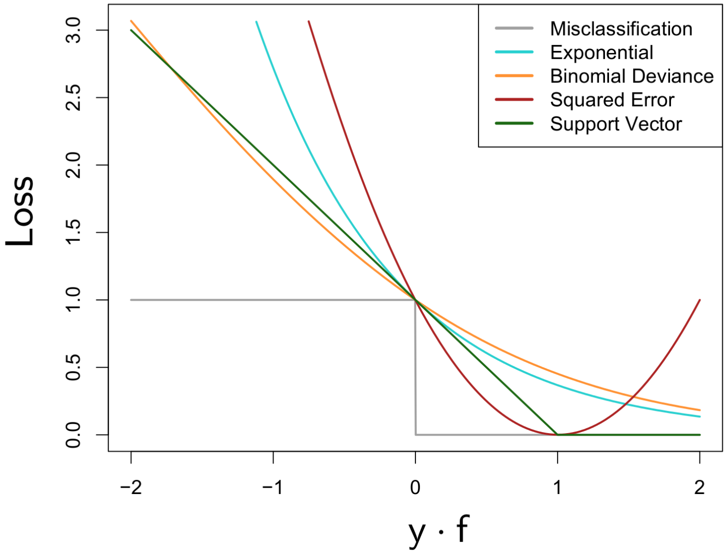
$$f_j(\mathbf{x}) \ = \ f_{j-1}(\mathbf{x}) + \beta_j \, f(\mathbf{x}, \gamma_j)$$

and hence

$$\begin{aligned} w_i^{(j+1)} \ &= \ w_i^{(j)} \, \exp\left(-\beta_j \, y_i \, f(\mathbf{x}_i, \gamma_j)\right) \\ &= \ \exp\left(-\beta_j\right) \, w_i^{(j)} \, \exp\left(-\alpha_j \, I\left(y_i \neq f(\mathbf{x}_i, \gamma_j)\right)\right) \end{aligned}$$

where $\alpha_j = 2 \, \beta_j$

Loss functions



Boosting

- The exponential loss penalizes misclassifications more than it approves correct classifications
- In particular, severe mistakes are very costly
- but the benefit of correct calls changes much more slowly

Boosting

- One can show that the "population" solution

$$\arg\min_{G(\mathbf{x})} E_{Y|\mathbf{X}=\mathbf{x}} \left[\exp\left(-Y\, G(\mathbf{x})\right)\right] \;=\; = \frac{1}{2} \log\left(\frac{P\left(Y = 1 | \mathbf{X} = \mathbf{x}\right)}{P\left(Y = -1 | \mathbf{X} = \mathbf{x}\right)}\right)$$

- The deviance loss also has the same "target" solution but grows slower - (so what?)

Boosting

- The shape of the exponential loss means that even if we have perfect classification for the training data, the objective function

$$\frac{1}{n}\sum_{i=1}^{n} L\left(y_i, G(\mathbf{x}_i)\right)$$

  may not have reached its minimum

- Thus the iterations continue...

Boosting

- Since we know what this method is estimating

$$\frac{1}{2}\,\log\left(\frac{P\,(Y=1|\,\mathbf{X}=\mathbf{x})}{P\,(Y=-1|\,\mathbf{X}=\mathbf{x})}\right)$$

... and we know what type of functions is attempting to use

$$\sum_{j=1}^{K}\beta_j\,f(\mathbf{x},\gamma_j)\;=\;\frac{1}{2}\,\log\left(\frac{P\,(Y=1|\,\mathbf{X}=\mathbf{x})}{P\,(Y=-1|\,\mathbf{X}=\mathbf{x})}\right)$$

Boosting

- ... we can understand when it works and when it may not work
- Note that the class of base classifiers $f(\mathbf{x}, \gamma)$ determines the type of log odds ratio we can model
- In particular, when using trees, the number of leaves (terminal nodes) determines the degree of interaction among the features that it may be able to capture

