The Boosting Approach to Machine Learning An Overview (Robert E. Schapire)

Giulio Meneghin, Javier Kreiner

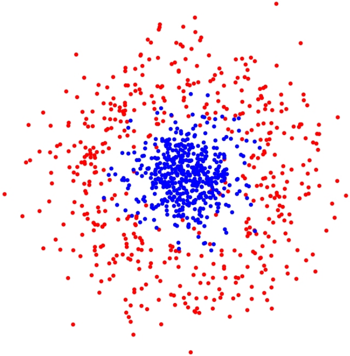
March 4, 2009

Boosting Approach: Combining simple rules

- Suppose we have a classification problem.
- Sometimes finding a lot of simple but not so accurate classification rules is much easier than finding a single highly accurate classification rule. (an algorithm that provides us with simple rules is called a **weak learner**)
- Many weak classifiers(simple rules) $\longrightarrow$ One strong classifier(accurate rule).
- Key idea: Give importance to misclassified data. How? Have a distribution over the training data.
- Finally: Find a good way to combine weak classifiers into general rule.

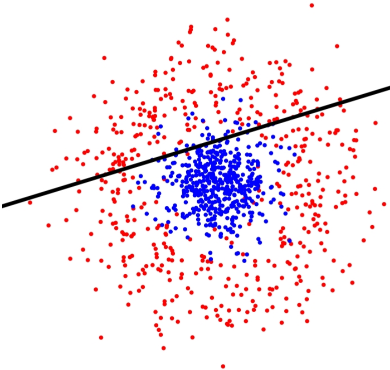
Example

Suppose we have points belonging to two different distributions.
Blue $\sim N(0,1)$, red $\sim \frac{1}{r\sqrt{8\pi^3}} e^{-\frac{1}{2}(r-4)^2}$



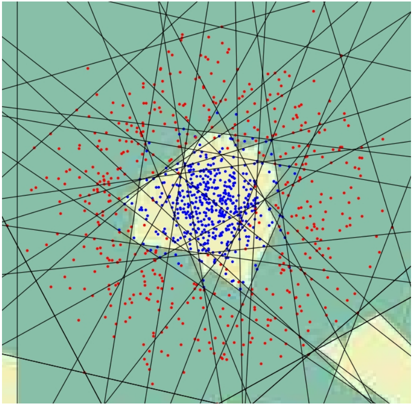
Example

It's very straightforward to come up with linear classifiers:



Example

If we *combined* many of this classifiers we could obtain a very accurate rule:



What is a weak learner?

- Given a set of examples $X = \{(x_1, y_1), ..., (x_m, y_m)\}$ with $y_i \in Y = \{-1, 1\}$, and a distribution over the examples $D_t(i)$, the weak learner gives us a hypothesis (or classifier)

$$h_t : X \to Y$$

- The goodness of that classifier is measured by the error:

$$\epsilon_t = \mathrm{Pr}_{i \sim D_t}[h_t(x_i) \neq y_i] = \sum_{i: h_t(x_i)_i} D_t(i)$$

- We ask of a weak learner to give consistently an error $\epsilon_t < \frac{1}{2} - \gamma$ (slightly less than chance) for any distribution over the examples.

NOTE: If the learning algorithm doesn't accept a distribution we just sample the training set according to D_t and provide the new sampled training set to the weak

Adaboost pseudocode

Input:

- set of examples $X = \{(x_1, y_1), ..., (x_m, y_m)\}$ with $y_i \in Y = \{-1, 1\}$
- a weak learning algorithm *WeakLearn*
- number of iterations T

Adaboost pseudocode

Initialize distribution $D_t(i) = \frac{1}{m}$ for all i (same weight for every example) For t = 1...T

1. Call *WeakLearn* using distribution D_t.

Adaboost pseudocode

Initialize distribution $D_t(i) = \frac{1}{m}$ for all i (same weight for every example) For t = 1...T

1. Call *WeakLearn* using distribution D_t.
2. Get back a classifier $h_t : X \rightarrow Y$

Adaboost pseudocode

Initialize distribution $D_t(i) = \frac{1}{m}$ for all i (same weight for every example) For t = 1...T

1. Call *WeakLearn* using distribution D_t.
2. Get back a classifier $h_t : X \rightarrow Y$
3. Calculate error of h_t, $\epsilon_t = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$. If $\epsilon_t > \frac{1}{2}$, then set T=t-1 and recommence loop

Adaboost pseudocode

Initialize distribution $D_t(i) = \frac{1}{m}$ for all i (same weight for every example) For t = 1...T

1. Call *WeakLearn* using distribution D_t.
2. Get back a classifier $h_t : X \rightarrow Y$
3. Calculate error of h_t, $\epsilon_t = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$. If $\epsilon_t > \frac{1}{2}$, then set T=t-1 and recommence loop
4. Set α_t. (For example $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$).

Adaboost pseudocode

Initialize distribution $D_t(i) = \frac{1}{m}$ for all i (same weight for every example) For t = 1...T

1. Call *WeakLearn* using distribution D_t.
2. Get back a classifier $h_t : X \rightarrow Y$
3. Calculate error of h_t, $\epsilon_t = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$. If $\epsilon_t > \frac{1}{2}$, then set T=t-1 and recommence loop
4. Set α_t. (For example $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$).
5. Update D_t:

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \\ e^{-\alpha_t y_i h_t(x_i)} & \text{, general formula} \end{cases}$$

where Z_t is a normalization constant (so as to generate a valid distribution), this procedure emphasizes difficult examples ($\alpha_t > 0$).

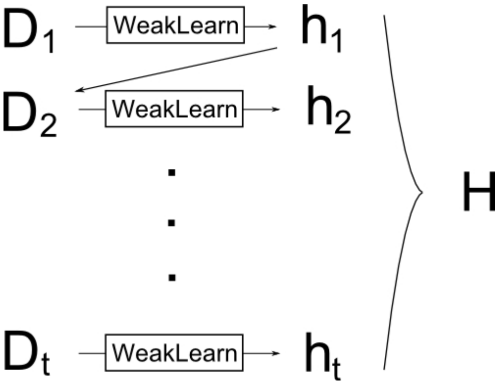
Adaboost pseudocode

Output: The final classifier:

$$h(x) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(x))$$

Note that this is a weighted voting of the classifiers of each iteration.

Diagram



Bound on the training error

Fortunately we have some strong guarantees for the training error (the proportion of misclassified samples):

$$\frac{1}{m}|\{i : H(x_i) \neq y_i\}| \leq \prod_t Z_t \leq e^{-2\sum_{t=1}^T \gamma_t^2} \leq e^{-2T\gamma^2}$$

Here $\gamma_t = \frac{1}{2} - \epsilon_t$. Provided *WeakLearn* does always better than chance so that $\gamma_t \geq \gamma > 0$:

- **The training error drops exponentially fast with T.**

Generalization error

The generalization error is bounded by:

$$\hat{Pr}[H(x) \neq y] + \tilde{O}(\sqrt{\frac{Td}{m}})$$

Where $\hat{Pr}[.]$ is the empirical probability, d is the Vapnik-Chervonenkis dimension, $\tilde{O}$ contains all the log and constant factors.

- **This suggests that there is overfitting?! (grows with T)** .
- **It doesn't often happen empirically, the bound is not tight enough.**
- **There other bounds for the generalization error, but in general give a qualitative measure. They are to weak to be quantitatively useful.**

Demos

http://vision.ucla.edu/~vedaldi/code/snippets/snippets.html

http://www.cse.ucsd.edu/~yfreund/adaboost/index.html

Extensions of AdaBoost

- There are a lot of variants of AdaBoost, depending on the specific problem to be solved.
- One of them is AdaBoost.M1.
- Used for k-class classification.

Algorithm AdaBoost.M1

Input: sequence of m examples $\langle(x_1, y_1), \ldots, (x_m, y_m)\rangle$
with labels $y_i \in Y = \{1, \ldots, k\}$
weak learning algorithm **WeakLearn**
integer T specifying number of iterations

Initialize $D_1(i) = 1/m$ for all i.

Do for $t = 1, 2, \ldots, T$:

1. Call **WeakLearn**, providing it with the distribution D_t.
2. Get back a hypothesis $h_t : X \to Y$.
3. Calculate the error of h_t: $\epsilon_t = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$.
   If $\epsilon_t > 1/2$, then set $T = t - 1$ and abort loop.
4. Set $\beta_t = \epsilon_t/(1 - \epsilon_t)$.
5. Update distribution D_t:
   $$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \beta_t & \text{if } h_t(x_i) = y_i \\ 1 & \text{otherwise} \end{cases}$$
   where Z_t is a normalization constant (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$h_{fin}(x) = \arg\max_{y \in Y} \sum_{t:h_t(x)=y} \log \frac{1}{\beta_t}.$$

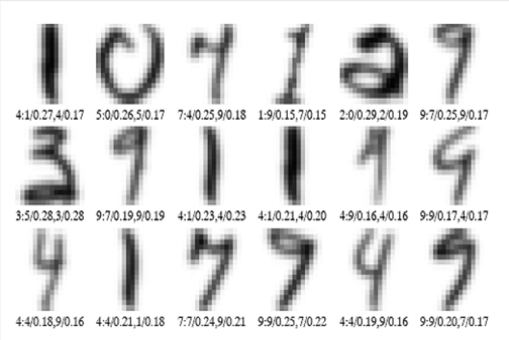
Multiclass classification: AdaBoost.M1

- If k $= 2$, then chance performance $= \frac{1}{2}$.
- If k $=$ n, then chance performance $= \frac{1}{n}$.
- AdaBoost.M1 still requires a WeakLearner performance of $\frac{1}{2}$.
- WeakLearner must be a strong learner if we want to use AdaBoost.M1.

Multiclass classification: AdaBoost.M2

- Instead of a class, WeakLearner can specify a set of plausible labels.
- The weak hypothesis is a k-element vector with elements in $[0, 1]$.
- $0 =$ not plausible, $1 =$ plausible ($\neq$ probable).
- Requires modification of WeakLearner.

OCR example


4:1/0.27,4/0.17
5:0/0.26,5/0.17
7:4/0.25,9/0.18
1:9/0.15,7/0.15
2:0/0.29,2/0.19
9:7/0.25,9/0.17
3:5/0.28,3/0.28
9:7/0.19,9/0.19
4:1/0.23,4/0.23
4:1/0.21,4/0.20
4:9/0.16,4/0.16
9:9/0.17,4/0.17
4:4/0.18,9/0.16
4:4/0.21,1/0.18
7:7/0.24,9/0.21
9:9/0.25,7/0.22
4:4/0.19,9/0.16
9:9/0.20,7/0.17

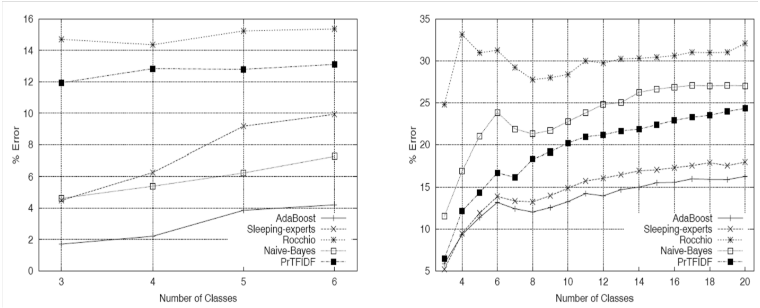
Incorporating Human Knowledge

- AdaBoost performance depends on the amount of data (among other things).
- A solution: exploit available human knowledge while boosting the WeakClassifier.

Incorporating Human Knowledge: BoosTexter

- Binary classification problem, classes in $\{-1, +1\}$
- Human expert constructs a probability function $p : D \rightarrow C$ that estimates the probability that an instance belongs to class $+1$.
- p needs not be highly accurate, another parameter μ is to control the confidence in the human expert knowledge.

Text classification performance



Comparison of Adaboost with four other text categorization methods in two datasets, Reuters newswire articles(left) and AP newswire headlines(right). x-axis: number of class labels.

Finding outliers

- AdaBoost gives higher weights to harder examples.
- Examples with highest weights are often outliers.
- If too many outliers, or too noisy data, performance decreases (although solutions proposed).

Applications

- **Text filtering** Schapire, Singer, Singhal. Boosting and Rocchio applied to text filtering.1998
- **Routing** Iyer, Lewis, Schapire, Singer, Singhal. Boosting for document routing.2000
- **Ranking problems** Freund, Iyer, Schapire, Singer. An efficient boostingalgorithm for combining preferences.1998
- **Image retrieval** Tieu, Viola. Boosting image retrieval.2000
- **Medical diagnosis** Merler, Furlanello, Larcher, Sboner. Tuning cost sensitive boosting and its application to melanoma diagnosis.2001

Conclusions

- Perspective shift: you may not need to find a perfect classifier, just combine a good enough classifier.
- An already seen example: face detection.
- Thoughts:
  - Fast, simple, easy to program.
  - Tuned by only one parameter (T, number of iterations).
  - Theoretical assurances, given:
    (1) WeakLearner performance,
    (2) Training set size.
  - Variants exist to address specific problems.

References

FREUND, Y.; SCHAPIRE, R.E., Experiments with a New Boosting Algorithm, Machine Learning: Proceedings of the Thirteenth International Conference, pp. 148-156, 1996

ROCHERY, M.; SCHAPIRE, R.E.; RAHIM, M.; GUPTA, N., BoosTexter for text categorization in spoken language dialogue, Unpublished manuscript, 2001

SCHAPIRE, R.E.; SINGER, Y., Improved boosting algorithm using confidence-rated predictions , Machine Learning, 37(3), pp.297-336, 1999

SCHAPIRE, R.E., The Boosting Approach to Machine Learning: An Overview, MSRI Workshop on Nonlinear Estimation and Classification, 2002

MATAS, J.; SOCHMAN, J. , AdaBoost, http://cmp.felk.cvut.cz

HOIEM, D., ADABOOST ,
www.cs.uiuc.edu/homes/dhoiem/presentations/Adaboost$_Tutorial.ppt$

Useful links

- http://www.boosting.org
- http://www.cs.princeton.edu/~schapire/boost.html

Thank you!