CS-E4710 Machine Learning: Supervised Methods

Lecture 9: Ensemble learning

Juho Rousu

November 16, 2021

Department of Computer Science
Aalto University

Ensemble learning

- Ensemble learning encompasses a wide range of machine learning frameworks which aim to construct high-performance predictive models by combining simpler base models
- Ensemble model generally predicts by computing an average or a majority vote over the base models, possibly weighted in some way
- Many approaches:
  - Boosting (classification task): incrementally add base models which focus on the mistakes made on the previous models by reweighing training examples
  - Bootstrap aggregation aka Bagging (classification or regression tasks): draw random subsamples with replacement from the original training data, train base classifer with each subsample, and combine by voting or averaging.
  - Gradient boosting (regression task): incrementally add base models that focus on the residuals of the prediction error

Why can model combination work? A thought experiment

Assume the following setup:

- A target concept $C : X \mapsto \{+1, -1\}$ to be learned
- To predict C, we have trained a collection of base hypotheses, binary classifiers $h_1, h_2, \ldots, h_L$, $h_j : X \mapsto \{-1, +1\}$
- We use a majority vote of the hypotheses as the prediction of the ensemble: $f_{maj}(\mathbf{x}) = \mathrm{argmax}_{y \in \mathcal{Y}} \sum_{j=1}^{L} \mathbf{1}_{h_j(\mathbf{x})=y}$, i.e. predict the label that agrees with the majority of the base hypotheses
- When L is even, both classes may have same number of votes, in that case tie breaking is needed, e.g. by predicting a random label
- This is a so called **majority voting ensemble**

Why can model combination work? A thought experiment

- Assume further that the base hypotheses have the same true risk ϵ but that the probabilities $P(h_j(\mathbf{x}) \neq C(\mathbf{x}))$ are independent between hypotheses for all h_i, h_j:

$$P(\{h_i(\mathbf{x}) \neq C(\mathbf{x})\} \text{ AND } \{h_j(\mathbf{x}) \neq C(\mathbf{x})\}) \\ = P(h_i(\mathbf{x}) \neq C(\mathbf{x})) \cdot P(h_j(\mathbf{x}) \neq C(\mathbf{x})) = \epsilon^2$$

- Not a practical assumption, used here for illustrating a phenomenon
- Now the probability of the majority vote $h(\mathbf{x})$ to be incorrect, i.e. the true risk $R(h)$ equals the probability of $k > L/2$ base hypotheses being incorrect, when L is odd (no tie-breaking needed)
- This probability is given by the tail of the cumulative binomial probability distribution:

$$R(h) = P(h(\mathbf{x}) \neq C(\mathbf{x})) = \sum_{k=\lceil L/2 \rceil}^{L} \binom{L}{k} \epsilon^k (1-\epsilon)^{L-k}$$

Why can model combination work? A thought experiment

Table shows the behaviour of the true risk of the ensemble with different values of L and ϵ

$$R(h) = P(h(\mathbf{x}) \neq C(\mathbf{x})) = \sum_{k=\lceil L/2 \rceil}^{L} \binom{L}{k} \epsilon^{k} (1-\epsilon)^{L-k}$$

- With low values of ϵ the risk goes down quickly to zero
- Even with $\epsilon = 0.45$ the risk is low with large enough L
- With $\epsilon = 0.5$ (i.e. random guessing) the risk remains at 0.5 independent of L

L	ϵ =0.1	0.33	0.45	0.5
5	0.0086	0.2050	0.4069	0.5000
11	0.0003	0.1171	0.3669	0.5000
101	0.0000	0.0002	0.1562	0.5000
501	0.0000	0.0000	0.0124	0.5000

Why can model combination work? A thought experiment

- Even if the the base classifers are **weak**, the combined classifer can be made accurate
- However, the assumption of independent errors is not a realistic one; in practise different classifiers trained on the same or similar data, tend to have correlated errors

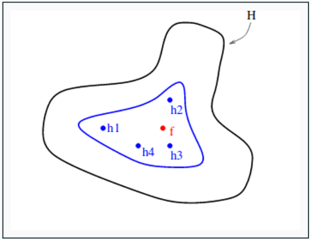
- In practice, similar kind of effect can be realized as long as the errors of the classifiers are not perfectly correlated
- A key question in **ensemble learning** is how to obtain **diverse** base classifiers, those that have different error patterns

L	$\epsilon =$0.1	0.33	0.45	0.5
5	0.0086	0.2050	0.4069	0.5000
11	0.0003	0.1171	0.3669	0.5000
101	0.0000	0.0002	0.1562	0.5000
501	0.0000	0.0000	0.0124	0.5000

Why can an ensemble work in practice?

Statistical reason (Dietterich, 2000):

- Consider learning as an optimization problem whose goal is to find the best $h \in \mathcal{H}$ in space of hypotheses $\mathcal{H}$.
- When the available data are not sufficient to identify the proper hypothesis, there might be several hypotheses with comparable accuracy.
- Averaging over all these hypotheses can limit the effect of selecting a suboptimal hypothesis.



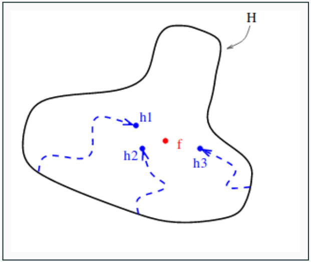
By combining models, we aim to "average out" some of the errors, as in our thought experiment

Dietterich, Thomas G. "Ensemble methods in machine learning." International workshop on multiple classifier systems. Springer, Berlin, Heidelberg, 2000.

Why can an ensemble work in practice?

Computational reason (Dietterich, 2000):

- The objective function might have several local minima to which the algorithm may converge for computational reasons (e.g. different starting point)
- The different locally optimal hypothesis would often make errors on different examples
- Combining the predictions of such suboptimal learners, can be beneficial to reduce the effect of getting stuck in a single local minimum

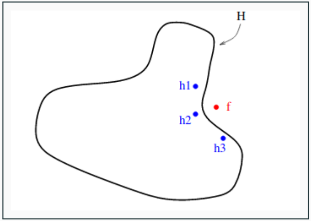


Dietterich, Thomas G. "Ensemble methods in machine learning." International workshop on multiple classifier systems. Springer, Berlin, Heidelberg, 2000.

Why can an ensemble work in practice?

Representational reason (Diettrich, 2000):

- The representational capacity of a hypothesis space can be extended beyond the search space by averaging over several hypotheses.
- For example: combination of hyperplanes let us represent sets of convex polygons, which is more general than sets of half-spaces, represented by single hyperplanes
- Thus an ensemble can learn patterns outside the original hypothesis class $\mathcal{H}$.



Dietterich, Thomas G. "Ensemble methods in machine learning." International workshop on multiple classifier systems. Springer, Berlin, Heidelberg, 2000.

Diverse learners

- To have an effective ensemble, it should consist of base hypotheses that are **diverse** in the sense that they make errors on different training examples
- Diversity among the hypotehesis can arise from different sourcest:
    - Algorithms: One can combine models of different types (e.g., neural networks, SVM's).
    - Hyperparameters:
        - different numbers of hidden units or layers in a multilayer perceptron
        - different kernel or regularization parameters in support vector machines
    - Input Data:
        - data fusion, where the data from different sources or measurement techniques are integrated (e.g. integrating natural light, infrared and X-ray images in astronomical data)
        - subsampling training data, by choosing random subsets from the examples, or using different input features

Diversity and model averaging

- Examine and ensemble for a regression task, generated by averaging the predictions of the base models

$$f_{avg}(\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T} h_t(\mathbf{x}), h_t(\mathbf{x}) \in \mathbb{R}, t = 1, \ldots, T$$

- We focus on its squared loss on a single example $\mathbf{x}$ that has true label $y \in \mathbb{R}$: $L_{sq}(f_{avg}(\mathbf{x}), y) = (f_{avg}(\mathbf{x}) - y)^2$
- We compare its performance to the average squared loss of the base models

$$\bar{L}_{sq}(\mathbf{x}, y) = \frac{1}{T}\sum_{t=1}^{T} L_{sq}(h_t(\mathbf{x}), y) = \frac{1}{T}\sum_{t=1}^{T}(h_t(\mathbf{x}) - y)^2$$

Brown, G. and Kuncheva, L.I., 2010, April. “Good” and “bad” diversity in majority vote ensembles. In International workshop on multiple classifier systems (pp. 124-133). Springer, Berlin, Heidelberg.

Diversity and model averaging

- The squared loss on the ensemble $f_{avg}(\mathbf{x})$ on a single example $\mathbf{x}$ with true label y satisfies (Brown and Kuncheva, 2010):

$$L_{sq}(f_{avg}(\mathbf{x}), y) = \bar{L}_{sq}(\mathbf{x}, y) - \frac{1}{T}\sum_{t=1}^{T}(h_t(\mathbf{x}) - f_{avg}(\mathbf{x}))^2$$

  - The first term on the right-hand side is the average loss of the base models
  - The second term represents the diversity of the base hypotheses in terms of their variance around the ensemble
  - As the variance is always non-negative, the ensemble error is always as good as the average error of the base models

Conclusion: for the averaging model using squared loss, diversity among the base models always helps

Brown, G. and Kuncheva, L.I., 2010, April. "Good" and "bad" diversity in majority vote ensembles. In International workshop on multiple classifier systems (pp. 124-133). Springer, Berlin, Heidelberg.

Diversity and majority voting

- Examine now a classification problems using a majority vote:

$$f_{maj}(\mathbf{x}) = \mathrm{argmax}_y \sum_{t=1}^{T} \mathbf{1}_{h_T(\mathbf{x})=y}$$

- We focus on the zero-one loss of the ensemble on $\mathbf{x}$ with true label $y \in \{-1, +1\}$:

$$L_{0/1}(f_{maj}(\mathbf{x}), y) = \mathbf{1}_{f_{maj}(\mathbf{x}) \neq y}$$

- We compare its loss to the average zero-one loss of the base models:

$$\bar{L}_{0/1}(\mathbf{x}, y) = \frac{1}{T} \sum_{t=1}^{T} L_{0/1}(f_t(x), y)$$

Brown, G. and Kuncheva, L.I., 2010, April. "Good" and "bad" diversity in majority vote ensembles. In International workshop on multiple classifier systems (pp. 124-133). Springer, Berlin, Heidelberg.

Diversity and majority voting

- Brown and Kuncheva (2010) showed that the loss of the majority voting ensemble can be expressed as:

$$L_{0/1}(f_{maj}(x), y) = \bar{L}_{0/1}(\mathbf{x}) - y f_{maj}(\mathbf{x}) \frac{1}{T} \sum_{t=1}^{T} \mathbf{1}_{h_t(\mathbf{x}) \neq f_{maj}(\mathbf{x})}$$

- The first term is the average zero-one error of the base models
- The second term accounts for the disagreements between the ensemble and the base models:
  - When the ensemble is correct ($y = f_{maj}(\mathbf{x})$) disagreements reduce the majority vote error
  - When the ensemble is incorrect ($y \neq f_{maj}(\mathbf{x})$) disagreements increase the majority vote error

Conclusion: For majority voting ensembles, diversity among the base models has context-dependent effects on the accuracy

Boosting

Boosting

- Boosting originally was proposed as an answer to a question raised in Probably Approximately Correct (PAC) theory: Can **weak learners**, whose accuracy is only slightly better than random guessing, can be combined into a strong PAC learner?
- This question was answered positively by Rob Shapire in 1990, a theoretical finding that led to the development of a very practical AdaBoost (Adaptive Boosting) algorithm in 1996, and to the prestigious Gödel prize to be awarded for Schapire and Yoav Freund in 2003.
- AdaBoost creates diversity by re-weighting the training examples during the algorithm, and conducts a weighted majority vote to predict.

Weak learning

A concept class C is **weakly PAC-learnable** if there exists an algorithm A, $\gamma > 0$, such that

- for all $\delta > 0$ for all $c \in C$ and all distributions D,

$$P_{S \sim D}(R(h_S) \leq \frac{1}{2} - \gamma) \geq 1 - \delta$$

- for a sample S of size m which is polynomial in $1/\delta$

Note: The definition differs from the PAC learnability by only requiring the true risk to be slightly less than random, with high confidence

Boosting theorem

Boosting Theorem (Schapire, 1990): A concept class C is weakly PAC-learnable if and only if it is (strongly) PAC-learnable.

- This surprising result implies that learning is an all or nothing phenomenon: if we can find an algorithm that achieves a low level of accuracy in learning C, then there exists an algorithm that can do the same with a high level of accuracy.
- AdaBoost algorithm described next is a practical algorithm that achieves a (strong) PAC learner by linear combinations of weak learners

Schapire, R.E., 1990. The strength of weak learnability. Machine learning, 5(2), pp.197-227.

AdaBoost

AdaBoost algorithm iteratively combines weak learners to arrive a strong classifier in the PAC sense

Inputs:

- A labeled training sample $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{m}$, $\mathbf{x}_i \in X$, $y_i \in \{-1, +1\}$
- A hypothesis class H where base hypotheses $h_t, t = 1, \ldots, T$ are drawn
- A distribution $D_t, t = 1, \ldots, T$, that assigns a weight $D_t(i)$ for the i'th training example for the t'th weak learner h_t

Output: combined model as a non-negative ($\alpha_t \geq 0$) combination of the weak learners

$$f_T(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})$$

Note the loose terminology: a weak learner is actually the algorithm outputting the base model, but in boosting literature the base models h_t are also called weak learners

AdaBoost

- In each round t, the algorithm adds a new weak learner h_t, one minimizes the empirical error on a sample drawn from distribution D_t (same as empirical errors weighted by D_t):

$$\epsilon_t = \min_{h \in H} P_{i \sim D_t}(h(\mathbf{x}_i) \neq y_i) = \min_{h \in H} \sum_{i=1}^{m} D_t(i) \mathbf{1}_{h(\mathbf{x}_i) \neq y_i}$$

- The weak learner h_t is weighted by

$$\alpha_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}$$

  which can be seen as log-odds probability of the weak learner being correct. This value comes from minimization of a bound for the empirical risk.

- If $\epsilon_t < 1/2$ then $\frac{1-\epsilon_t}{\epsilon_t} > 1$ and $\alpha_t > 0$ thus weak learners that are more accurate on training data than random guessing receive positive weights.

AdaBoost

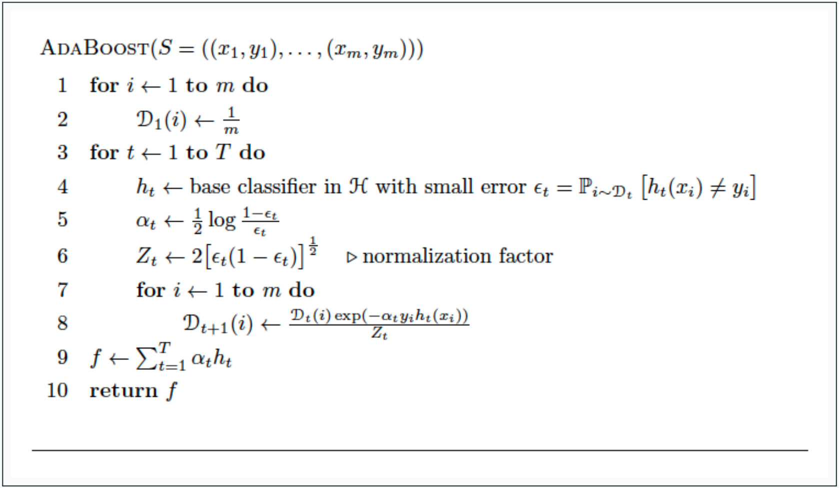
For the round $t+1$, the weights on the training examples are re-weighted:

$$D_{t+1}(i) = D_t(i) \cdot \frac{e^{-\alpha_t y_i h_t(\mathbf{x}_i)}}{Z_t}$$

- The term $y_i \alpha_t h_t(\mathbf{x})$ can be seen as a margin of example (weighted by α_t)
- Feeding the margin through an negative exponential causes an exponential up-weighting of misclassified examples (those with negative margin) and down-weighting of correctly classified examples (positive margin)
- The factor: $Z_t = \sum_{i=1}^{m} D_t(i) e^{-\alpha_t y_t h_t(\mathbf{x}_i)} = 2\sqrt{\epsilon_t(1-\epsilon_t)}$ is a normalization factor ensuring that D_t sums up to 1.

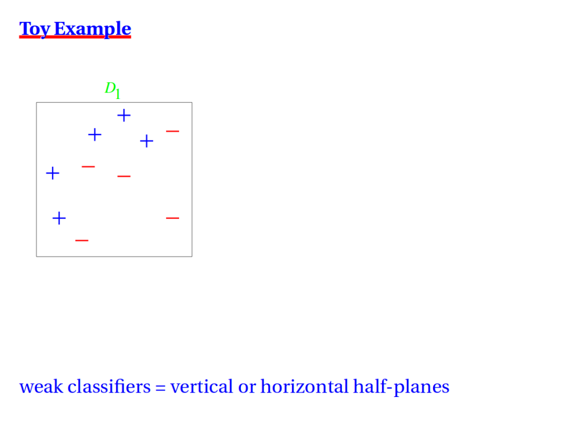
The sequence of the re-weighting is that the weak learners for subsequent rounds will focus on the examples that were misclassified ⇒ increases diversity

AdaBoost pseudo-code



```
$\textsc{AdaBoost}(S = ((x_1, y_1), \ldots, (x_m, y_m)))$
for $i \leftarrow 1$ to $m$ do
    $\mathcal{D}_1(i) \leftarrow \frac{1}{m}$
for $t \leftarrow 1$ to $T$ do
    $h_t \leftarrow$ base classifier in $\mathcal{H}$ with small error $\epsilon_t = \mathbb{P}_{i \sim \mathcal{D}_t}\left[h_t(x_i) \neq y_i\right]$
    $\alpha_t \leftarrow \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
    $Z_t \leftarrow 2\left[\epsilon_t(1-\epsilon_t)\right]^{\frac{1}{2}}$    ▷ normalization factor
    for $i \leftarrow 1$ to $m$ do
        $\mathcal{D}_{t+1}(i) \leftarrow \frac{\mathcal{D}_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$
$f \leftarrow \sum_{t=1}^{T} \alpha_t h_t$
return $f$
```

AdaBoost example

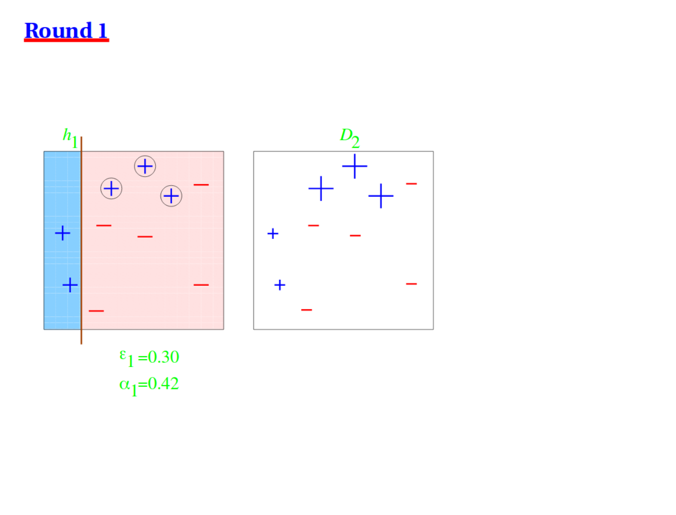


Toy Example

weak classifiers = vertical or horizontal half-planes

source : *https : / / www . csie . ntu . edu . tw / mhyang / course / u0030 / papers / schapire . pdf*

AdaBoost example



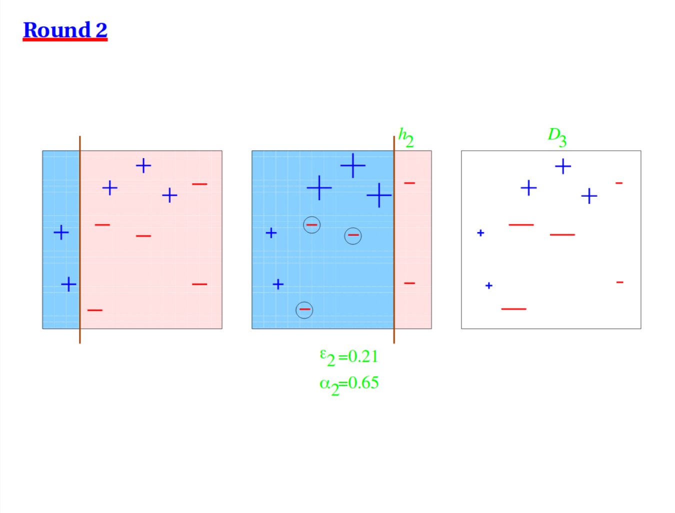
Round 1

$\varepsilon_1 = 0.30$

$\alpha_1 = 0.42$

source: *https://www.csie.ntu.edu.tw/ mhyang/course/u0030/papers/schapire.pdf*

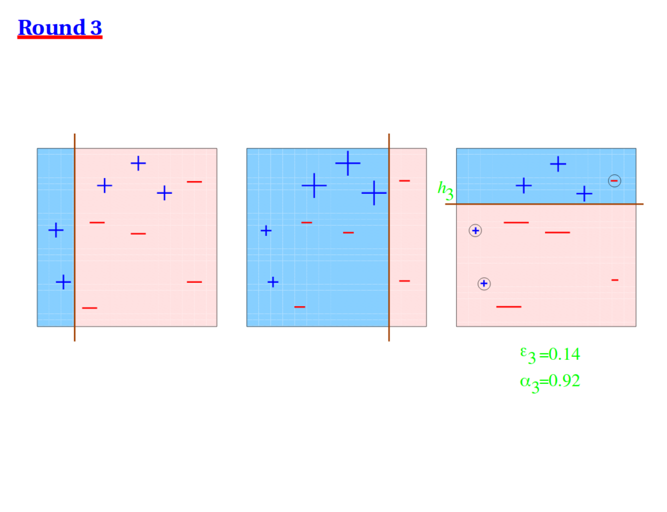
AdaBoost example



Round 2

source: *https : / / www . csie . ntu . edu . tw / mhyang / course / u0030 / papers / schapire . pdf*

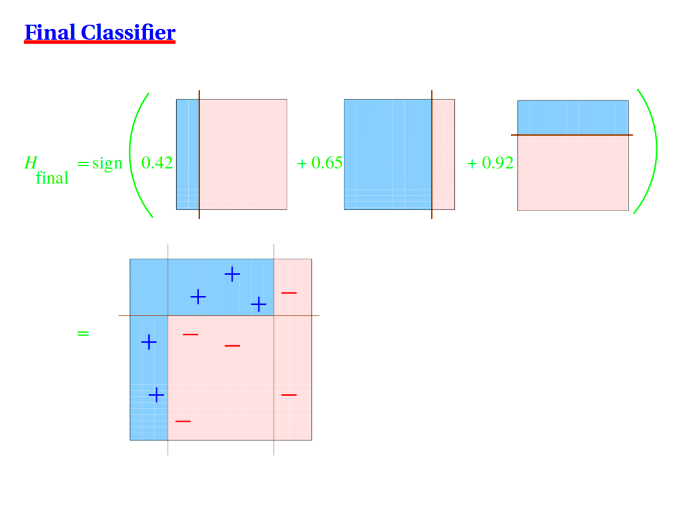
AdaBoost example



Round 3

source: *https : / / www . csie . ntu . edu . tw / mhyang / course / u0030 / papers / schapire . pdf*

AdaBoost example



Final Classifier

$$H_{\text{final}} = \text{sign}\left(0.42 \qquad + 0.65 \qquad + 0.92 \qquad \right)$$

$$=$$

source: *https : / / www . csie . ntu . edu . tw / mhyang / course / u0030 / papers / schapire . pdf*

Empirical error of AdaBoost

Theorem: The empirical error of the classifier f returned by AdaBoost verifies:

$$\hat{R}_S(f) \leq \exp\left(-2\sum_{t=1}^{T}\left(\frac{1}{2} - \epsilon_t)^2\right)$$

Furthermore, if for all t, $\gamma \leq (\frac{1}{2} - \epsilon_t)$, then

$$\hat{R}_S(f) \leq \exp(-2\gamma^2 T)$$

Proof: See Mohri book

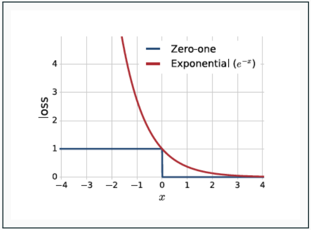
- The empirical error goes down exponentially fast in T: given enough weak learners, empirical error can be pushed arbitrarily low
- Above γ is the "edge", the amount by which the weak learners are more accurate than random guessing: larger γ results in tighter bound for the empirical error

The loss function optimized by AdaBoost

It can be shown that AdaBoost is minimizing an upper bound on the zero-one loss, given by the exponential function

$$F_{exp}(\boldsymbol{\alpha}) = \sum_{i=1}^{m} e^{-y_i f_T(\mathbf{x}_i)} = \sum_{i=1}^{m} e^{-y_i \sum_{t=1}^{T} \alpha_t h_t(x_i)}$$

- Here the variables to be optimized are the weights $\boldsymbol{\alpha} = (\alpha_t)$, and the collection of base learners are taken as fixed
- Exponential loss penalizes misclassified examples heavily, which revels a potential weakness of AdaBoost: if labels are noisy, AdaBoost will give high weight on noisy labels with the risk of overfitting



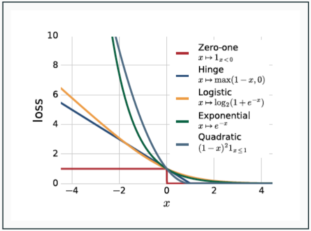
The loss function optimized by AdaBoost

The viewpoint of loss function optimization suggest alternative boosting algorithms by changing the loss function

- For example, LogitBoost (Friedman et al. 2000) minimizes the logistic loss:

$$F_{logit}(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \log(1+e^{-y_i \sum_{t=1}^{T} \alpha_t h_t(x_i)})$$

- LogitBoost is closely related logistic regression (LR): if we consider the weak learners h_t as the input features and α_t as the feature weights, LogitBoost is essentially learning a logistic regression model (except for some constants that differ from LR).



Friedman, J., Hastie, T., Tibshirani, R. (2000). "Additive logistic regression: a statistical view of boosting". Annals of Statistics. 28 (2): 337–407

VC-dimension of AdaBoost

- The hypothesis class of AdaBoost after T rounds is given by

$$\mathcal{F}_T = \left\{ sgn(\sum_{t=1}^{T} \alpha_t h_t) : \alpha_t > 0, h_t \in H) \right\}$$

- The VC-dimension of $\mathcal{F}_T$ can be bounded in terms of the VC-dimension d of the weak learners:

$$VCdim(\mathcal{F}_T) \leq 2(d+1)(T+1)\log_2((T+1)e)$$

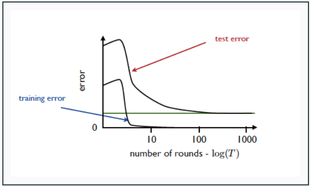
  which grows as $O(dT \log T)$

- This suggests that AdaBoost could overfit when T grows large, however empirically it has been observed not to be the case

AdaBoost and generalization

Empirically AdaBoost has been observed not to overfit with large T

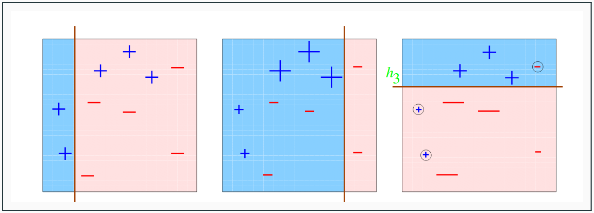
- Instead, the test error continues to go down with increasing T even after the empirical error reaches zero
- This phenomenon has been explained through margin-based analysis of AdaBoost:



- AdaBoost can be shown to increase the margins of training examples as T increases, which leads to better generalization
- However, AdaBoost is not strictly speaking optimizing the margins, but can achieve substantial fraction of the maximum margin in suitable conditions
- Boosting algorithms that aim to maximize margins have been proposed, however, they do not empirically outperform AdaBoost

Boosting in practice

- Boosting requires having an access to a weak learner, it is hard to guarantee formally in real world situations
- However, empirically "reasonable" classifiers generally outperform random guessing on a real world data
- In general a weak learner that is unstable, that is, the chosen hypothesis changes upon small modification of data, works the best (due to introducing diversity to the ensemble)
- In practice, axis-paraller hyperplanes, also called "decision stumps" work well



Other ensemble learning schemes

Bootstrap Aggregating (Bagging)

- Bootstrap aggregating (Bagging) is an ensemble method in which the weak learners are built using T different bootstrap samples of the original training data.
- Bootstrap sampling : drawing randomly with replacement from the original training set (resampling with replacement) m instances, where m is the size of the training data.
- The bootstrap samples are different from each other, which creates diversity
- For classification, the ensemble prediction is given by majority voting
- **Random forest** is an effective ensemble learning methods that combines randomized decision trees with bagging

Gradient boosting

- Gradient boosting (GB) (Friedman, 2011) is an ensemble method based on iteratively growing the ensemble according to the gradient of the loss function

$$f_t(\mathbf{x}) = f_{t-1}(\mathbf{x}) - \eta_t \sum_{i=1}^{m} \nabla_{f_{t-1}} L(y_i, f_{t-1}(\mathbf{x}_i))$$

- The partial derivatives composing the negative gradient $-\nabla_{f_{t-1}}$ are called the pseudo-residuals

$$r_{it} = -\frac{\partial L(y_i, f_{t-1}(\mathbf{x}))}{\partial f_{t-1}}$$

- In each iteration, a weak learner h_t is trained to predict the pseudo-residuals r_{it} and added to the ensemble: with a step-size η_t that is optimized for maximal descent

Friedman, J. (2001). Greedy boosting approximation: a gradient boosting machine. Ann. Stat. 29, 1189–1232. doi: 10.1214/aos/1013203451

Summary

- Ensemble methods are based on the idea of generating strong predictive models by combining simpler, potentially weaker models (weak learners)
- Diversity of the weak learners is the key for obtaining an accurate ensemble
- Several mechanisms exist for obtaining diversity
- Boosting is a successful ensemble method that is based on adaptively reweighting training examples