Boosting is an _______ method.

Unlike bagging or random forests, boosting determines a _______ majority vote.

In particular, if the class labels are $y = +1, -1$, then boosting determines

$$f_1, \ldots, f_T$$ _______ classifiers

$$\alpha_1, \ldots, \alpha_T > 0$$ weights

and returns the ensemble rule

$$h_T(x) =$$

Intuitively, $\alpha_t$ reflects the _______ in $f_t$. 
Let training data \((x_1, y_1), \ldots, (x_n, y_n)\) be fixed. Let \(F\) be a fixed set of classifiers called the ______ class.

**Definition** A ______ for \(F\) is a rule that takes as input a set of weights \(w_1, \ldots, w_n\) \((w_i \geq 0, \sum_{i=1}^{n} w_i = 1)\) and returns a classifier \(f \in F\) such that

is minimized (or at least small)

**Notation** A set of weights will be expressed as a vector:

\[
\mathbf{w} = (w_1, \ldots, w_n)
\]
The Boosting Principle

Choose an initial weight vector $w_1$.
Fix $T$ and a base class $F$.
For $t = 1:T$
  • Given the weight
    the _______ to generate
    a classifier $f_t$
  • Determine a confidence $\alpha_t > 0$
    in $f_t$
    • If $f_t(x_i) \neq y_i$, then
      $w_i$ w$i$
    • If $f_t(x_i) = y_i$, then
      $w_i$ w$i$
End
Output

$h_T(x) =$
Examples of base classes

- Decision trees

\[ f(x) = \pm \text{sign} \left\{ \sum x^{(i)} - c \right\} \]

- Decision (trees of depth )

- Radial basis functions

\[ f(x) = \pm \text{sign} \left\{ k_\sigma (x-x_i) - b \right\} \]

where \( k_\sigma \) is a radially symmetric .

Recall the advantages of ensemble methods:

- increased stability (decision trees)

- combine simple classifiers (stumps, RBFs)

\underline{Adaboost}

\( \rightarrow \) the first successful boosting algorithm, introduced by Yoav Freund and Robert Schapire.
Given \((x_1, y_1), \ldots, (x_n, y_n)\), \(y_i \in \{-1, +1\}\)

Initialize \(w_i^1 = \frac{1}{n}\).

For \(t = 1, \ldots, T\)

- Apply base learner with weights \(w^t\) to produce classifier \(f_t^t\)

- Set \(r_t = \sum_{i=1}^{n} w_i^t \mathbb{1}\{f_t(x_i) = y_i\}\)

- Set \(a_t = \frac{1}{2} \ln \left( \frac{1 - r_t}{r_t} \right)\)

- Update \(w_i^{t+1} = \frac{w_i^t \cdot \exp \left\{ - a_t y_i f_t(x_i) \right\}}{Z_t}\)
  where \(Z_t\) is a normalization constant

End

Output \(h_T(x) = \text{sign} \left\{ \sum_{t=1}^{T} a_t f_t(x) \right\}\)
Weak learning

The success of Adaboost is reflected in the following result.

Denote $\delta_t = \frac{1}{2} - \frac{1}{2} t$. We may assume $\delta_t \geq 0$ (why?)

Theorem: The training error of Adaboost satisfies

$$\frac{1}{n} \sum_{i=1}^{n} \frac{1}{2} \delta_t \left( h_t(x_i) \neq y_i \right) \leq \exp \left( -2 \sum_{t=1}^{T} \delta_t^2 \right)$$

In particular, if $\delta_t \geq \delta > 0$ for all $t$, then

$$\frac{1}{n} \sum_{i=1}^{n} \frac{1}{2} \delta_t \left( h_t(x_i) \neq y_i \right) \leq$$
The assumption $\delta_t \geq \delta > 0 \ \forall t$ is called the ______ ________

and in this case the base learner is called a ______ learner.

In words, the theorem tells us that if our base learner does slightly better than ______ ________, the final ensemble classifier can separate the training data perfectly for $T$ large enough. In fact the training error goes to zero ______ ________.
If $r = 0$, then $\alpha_t$ =
Does this make sense?

Setting $T$ is a ________ problem. If $T$ is too large we may experience ________.

Cross-validation is a common approach.

Empirical results suggest that AdaBoost with decision trees is one of the best "off-the-shelf" methods for classification.
Proof of Theorem

The proof is broken down into some lemmas.

Lemma

\[
\frac{1}{n} \sum_{i=1}^n \mathbf{1}_{h^T_T(x_i) \neq y_i}^2 \leq \prod_{t=1}^T Z_t
\]

Proof

By unraveling the update rule we find

\[
W_i^{T+1} = \frac{W_i^T \exp \left( -a_t y_i \frac{f_t(x_i)}{Z_T} \right)}{Z_T}
\]

\[
= \frac{W_i^{T-1} \exp \left( -y_i \left[ a_{T-t} f_{T-t}(x_i) + a_T f_T(x_i) \right] \right)}{Z_{T-1} \cdot Z_T}
\]

\[
= \left( \frac{1}{n} \cdot \exp \left( -y_i \sum_{t=1}^T a_t f_t(x_i) \right) \right)
\]

\[
\rightarrow \frac{\exp \left( -y_i F_T(x_i) \right)}{n \prod_{t=1}^T Z_t}
\]

where

\[
F_t = \sum_{s=1}^t \kappa_s f_s
\]

and

\[
h_t(x) = \text{sign} \left\{ F_t(x) \right\}
\]
Now use the bound

$$\frac{1}{\delta_t(x_i) \neq y_i^3} = \frac{1}{\frac{1}{\delta_t} y_i F_t(x_i) < 0^3} \leq \exp \left( -y_i F_t(x_i) \right)$$

Then

$$1 = \sum_{i=1}^{n} \exp(-y_i F_t(x_i))$$

$$= \sum_{i=1}^{n} \frac{\exp(-y_i F_t(x_i))}{n \cdot (\pi Z_t)}$$

$$\geq \frac{1}{(\pi Z_t)} \cdot \frac{1}{n} \sum_{i=1}^{n} \frac{1}{\delta_t(x_i) \neq y_i^3}$$

and the lemma follows.
Lemma \[ Z_t = \sqrt{1 - 4\delta_t^2} \]

Proof \[ Z_t = \sum_{i=1}^{n} \frac{w_i^t \exp \left(-\alpha_t y_i f_t(x_i) \right)}{w_i^{t+1}} \]

\[ = \sum_{i: f_t(x_i) = y_i} w_i^t \exp(-\alpha_t) + \sum_{i: f_t(x_i) \neq y_i} w_i^t \exp(\alpha_t) \]

\[ = (1-r_t) e^{-\alpha_t} + r_t e^{\alpha_t} \]

Now recall \[ \alpha_t = \frac{1}{2} \ln \left( \frac{1-r_t}{r_t} \right) = \ln \sqrt{\frac{1-r_t}{r_t}} \]

Then \[ Z_t = (1-r_t) \sqrt{\frac{r_t}{1-r_t}} + r_t \sqrt{\frac{1-r_t}{r_t}} \]

\[ = 2 \sqrt{r_t (1-r_t)} \]

Now substitute \[ r_t = \frac{1}{2} - \delta_t \]
\[ \Rightarrow z_t = 2 \sqrt{\left( \frac{1}{2} - \delta_t \right) \left( \frac{1}{2} + \delta_t \right)} \]
\[ = 2 \sqrt{\frac{1}{4} - \delta_t^2} \]
\[ = \sqrt{4 \delta_t^2} \]

Lemma

\[ \sqrt{1 - \kappa} \leq e^{-\frac{1}{2} \kappa} \]

Proof
Formally, $\sqrt{1-x}$ is concave, $e^{\frac{1}{2}x}$ is convex, so it suffices to show their slopes (derivatives) are both $= -\frac{1}{2}$ at $0$.

Putting it all together, we obtain

$$\frac{1}{n} \sum_{i=1}^{n} I_{\delta_{i}^{2}(x_{i}) \neq y_{i}^{2}} \leq \frac{1}{n} \sum_{i=1}^{n} \exp(-y_{i}(F_{i}(x_{i}))$$

$$= \prod_{t=1}^{T} Z_{t}$$

$$= \prod_{t=1}^{T} \sqrt{1-y_{i}^{2}}$$

$$\leq e^{-2 \sum_{t=1}^{T} \gamma_{t}^{2}}$$

Exercise: View $Z_{t}$ as a function of $\gamma_{t}$, and find the value of $\gamma_{t}$ that minimizes $Z_{t}$.
Solution

Earlier we showed

\[ Z_t = (1 - r_t) e^{-\alpha_t} + r_t e^{\alpha_t} \]

This is a convex, differentiable function of \( \alpha_t \).

It is minimized by setting

\[ 0 = \frac{\partial Z_t}{\partial \alpha_t} = -(1 - r_t) e^{-\alpha_t} + r_t e^{\alpha_t} \]

\[ \Rightarrow e^{2\alpha_t} = \frac{1 - r_t}{r_t} \]

\[ \Rightarrow \alpha_t = \frac{1}{2} \ln \left( \frac{1 - r_t}{r_t} \right) \]

In conclusion, each \( \alpha_t \) is chosen to minimize the corresponding term \( Z_t \) in the bound \( \sum_{t=1}^{T} Z_t \).

That is, the bound is minimized \underline{incrementally} (not \underline{globally}).
Boosting as Functional Gradient Descent

We will now generalize Adaboost to an entire class of boosting algorithms.

Recall that in bounding the Adaboost training error we used the bound

\[ \frac{1}{\xi} u < 0 \]

\[ u = y_i F_t(x_i) \]

We will generalize Adaboost by using the bound

\[ \frac{1}{\xi} u < 0 \]

where \( \phi(u) \) is called a _______ function.
For computational reasons, the loss is often chosen to be

Examples

- exponential
- logistic
- hinge (not differentiable, decreasing)
- squared error (not decreasing)

Why use different losses?

The logistic loss, for example, doesn't penalize misclassified points as severely, and therefore may be less susceptible to ___________.

Let's assume that $\phi$ is convex, differentiable, and decreasing.
On the $t^{th}$ iteration of boosting we have the ensemble

$$F_t(x) = \sum_{s=1}^{t} \alpha_s f_s(x)$$

and the bound

$$\frac{1}{n} \sum_{i=1}^{n} \frac{1}{2} y_i \cdot F_t(x) < 0 \Rightarrow \frac{1}{n} \sum_{i=1}^{n} \phi(y_i \cdot F_t(x_i))$$

View this bound as an objective function to be minimized with respect to $F_t$.

Boosting may be viewed as functional gradient descent applied to the bound.

In particular, suppose $\alpha_1, f_1, \ldots, \alpha_{t-1}, f_{t-1}$ are given, and set

$$B_t(\alpha, f) = \frac{1}{n} \sum_{i=1}^{n} \phi(y_i \cdot F_{t-1}(x) + y_i \cdot \alpha f(x_i))$$
Then set

1) \( \frac{f}{t} = \text{function } f \in \mathcal{F} \) (base class) for which the directional derivative of \( B_t \) in the direction \( f \) is minimized.

2) \( \alpha_t = \text{stepsize } \alpha > 0 \) in the direction \( f_t \) for which \( B(\alpha, f_t) \) is minimized.

Step 1] The directional derivative of \( B_t \) in the direction \( f \) is

\[ \frac{\partial B_t(\alpha, f)}{\partial \alpha} \bigg|_{\alpha=0} = \]
Minimizing this is equivalent to minimizing

\[- \sum_{i=1}^{n} y_i f(x_i) \cdot \frac{\phi'(y_i f_{t-1}(x_i))}{\sum_{j=1}^{n} \phi'(y_j f_{t-1}(x_j))}\]

since \( \phi' < 0 \)

\[w^t_i\]

\[= \sum_{i=1}^{n} w^t_i \frac{1}{\sum_{i=1}^{n} \frac{1}{\phi'(y_i f(x_i) \neq y_i)}} - \sum_{i=1}^{n} w^t_i \frac{1}{\sum_{i=1}^{n} \frac{1}{\phi'(y_i f(x_i) = y_i)}}\]

\[= 2 \left( \frac{\sum_{i=1}^{n} w^t_i \frac{1}{\phi'(y_i f(x_i) \neq y_i)}}{\sum_{i=1}^{n} \frac{1}{\phi'(y_i f(x_i) = y_i)}} \right) - 1\]

To minimize this expression with respect to \( f \),

we can use the _______ .

\[\text{Step 2}\]

\[\alpha_t := \arg \min_{\alpha} B_t(\alpha, f_t)\]

\[= \arg \min_{\alpha} \frac{1}{n} \sum_{i=1}^{n} \phi(y_i f_{t-1}(x_i) + y_i \alpha f_t(x_i))\]
Given \((x_i, y_i), \ldots, (x_n, y_n)\), \(y_i \in \{-1, 1\}\), convex loss \(\phi\)

Initialize \(w_i^1 = \frac{1}{n}\)

For \(t = 1, \ldots, T\)

- Apply base learner with weights \(w^t\) to produce classifier \(f_t^\star\)

- Set

  \[
  \alpha_t = \arg \min_{\alpha} \frac{1}{n} \sum_{i=1}^{n} \phi \left( y_i f_{t-1}(x_i) + y_i \alpha f_t(x_i) \right)
  \]

- Update

  \[
  w_i^{t+1} = \frac{\phi'(y_i f_t(x_i))}{\sum_{j=1}^{n} \phi'(y_j f_t(x_j))}
  \]

End

Output

\[
\hat{h}_T(x) = \text{sign} \left\{ \frac{1}{T} \sum_{t=1}^{T} \alpha_t f_t(x) \right\} = \text{sign} \left\{ \sum_{t=1}^{T} \alpha_t f_t(x) \right\}
\]
Since $\phi$ is convex, $x$ is the solution of a convex, univariate optimization problem and can be found efficiently using Newton's method.

When $\phi(u) = e^{-u}$, the algorithm simplifies to AdaBoost. In this case

the algorithm simplifies to AdaBoost. In this case, $w_t$ has a nice formula since

\[ \phi'(a+b) = \phi'(a) \cdot \phi'(b) \]

$\alpha_t$ has a closed form solution.

When $\phi(u) = \log_2 (1 + e^{-2u})$, the algorithm is called AdaBoost. For computational efficiency, Friedman, Hastie, and Tibshirani suggest using only one step of Newton's method at each round.

Why did we assume $\phi$ to be decreasing?
A. ensemble, weighted, base

\[ h_t(x) = \text{sign} \left\{ \sum_{t=1}^{T} \alpha_t f_t(x) \right\}, \quad \text{confidence} \]

B. base, base learner, \[ \sum_{t=1}^{T} \alpha_t f_t(x) \]

C. base learner, increase, decrease

\[ h_t(x) = \text{sign} \left\{ \sum_{t=1}^{T} \alpha_t f_t(x) \right\} \]

D. + increased stability, performance, -: lose interpretability

stumps, I, kernel

E. exp \((-2\gamma^2T)\) F. weak learning hypothesis,

weak, random guessing, exponentially fast

F. \(\alpha\), model selection, overfitting

G. \(e^{-\alpha}\), \(\phi(u)\), loss

H. convex, differentiable, decreasing; outliers

I. \[ \frac{1}{n} \sum_{i=1}^{n} y_i f(x_i) - \phi'(y_i f_{t-1}(x_i)) \]

J. base learner

K. recursive, LogitBoost

L. So that \(\phi' < 0\) (Step 1)