Boosting

CAP6676: Knowledge Representation
Instructor: Guo-Jun Qi
Weak classifiers

- Weak classifiers
  - Decision stump – one layer decision tree
  - Naive Bayes – A classifier without feature correlations
  - Linear classifier – logistic regression

- Weak classifiers usually have larger training error but smaller variance.

- A single weak classifier is usually not adequate in real applications, but it is possible to combine an ensemble of weak classifiers to build a strong one.
Idea: weighted voting

• Combining an ensemble of weak classifiers by weighted voting
  • Learning an ensemble of weak classifiers
    • Although each weak classifier is not adequate of classifying the whole feature space, it can still output good result on certain parts of feature spaces.
  • Each weak classifier is given a weight based on its performance
    • More adequate classifier will vote with more weight.
    • Weighted voting usually generates better performance by combining complementary classifiers good at classifying different parts of feature spaces.

Combined classifier: \( f(x) = \text{sign} \left( \alpha_1 h_1(x) + \alpha_2 h_2(x) \right) \)
Problems to solve

• How are an ensemble of weak classifiers learned?
  • Decide which part of training samples each weak classifier focuses on.
• How is the weight of each weak classifier decided?
Boosting [Schapire 98’]

- Idea: learning a pool of weak classifiers (usually of the same type e.g., stump, logistic regression), on different sets of training examples resampled from different parts of an original training set.
Boosting – The Algorithm

• On each iteration t:
  • Weight each training example by how correctly it is classified so far
  • Learn a weak classifier $h_t$ that best classifies the weighted training examples.
    • New weak classifier should focus on those difficult examples
  • Decide a weight for this weak classifier $\alpha_t$

• Final classifier: $f(x) = \text{sign}(\sum_t \alpha_t h_t(x))$
Learning from Weighted Training Examples

• Consider a weighted dataset
  • $D(i)$ – weight of $i$-th training example $(x^{(i)}, y^{(i)})$
  • Interpretations:
    • $i$-th example is resampled from training set by weight $D(i)$

• Two ways to learn a weak classifier from weighted training examples
  • Resampling the training set by $D(i)$, and train a weak classifier from the resampled set
  • Learn a weak classifier directly from the weighted samples, e.g., a weighted logistic regression classifier
    \[
    h = \min_h \sum_i D(i) \text{loss}(h(x^{(i)}), y^{(i)})
    \]
AdaBoost [Freund & Shapire’ 95]

Given: \((x_1, y_1), \ldots, (x_m, y_m)\) where \(x_i \in X, y_i \in Y = \{-1, +1\}\)

Initialize \(D_1(i) = 1/m.\)

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

- Train weak learner using distribution \(D_t.\)
- Get weak classifier \(h_t : X \rightarrow \mathbb{R}.\)
- Choose \(\alpha_t \in \mathbb{R}.\)
- Update:

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

where \(Z_t\) is a normalization factor

Output final classifier

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

Initially equal weights

Naïve bayes, decision stump

Magic (+ve)

Increase weight if wrong on pt i

\(y_i h_t(x_i) = -1 < 0\)
Decide the combination weight for each weak classifier

• Weight of weak classifier $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$

Where $\epsilon_t$ is the weighted training error

$$\epsilon_t = \sum_i D(i) \delta(h_t(x_i) \neq y_i)$$

If a classifier is better than a random guess, $\epsilon_t < 0.5$, and $\alpha_t > 0$; otherwise, $\alpha_t < 0$. For the latter case, it is an adverse classifier rather than a weak classifier.
Boosting Example (Decision Stump)

• Three weak classifier
Boosting Example (Decision Stump)

- Final classifier

\[
H_{\text{final}} = \text{sign}(0.42 + 0.65 + 0.92)
\]
How good can the boosting reduce the training error?

• If each weak classifier $h_t$ is slightly better than random guess with $\epsilon_t < 0.5$, then the training error of Adaboost can be reduced exponentially fast in the number of weak classifiers combined $T$,

$$\frac{1}{m} \sum_{i=1}^{m} \delta(H(x_i) \neq y_i) \leq \exp \left( -2 \sum_{t=1}^{T} (1/2 - \epsilon_t)^2 \right)$$

Training Error

• It is astounding result as AdaBoost can reduce the training error to arbitrarily close to zero.
Proof: Training error of AdaBoost

• Note the fact that exponential function $\exp(-y_if(x_i))$ is an upper bound of the 0/1 loss $\delta(y_i \neq f(x_i))$ as
Proof: Training error of AdaBoost

• The total training error is bounded by

\[ \frac{1}{m} \sum_{i=1}^{m} \delta(H(x_i) \neq y_i) \leq \frac{1}{m} \sum_{i=1}^{m} \exp(-y_i f(x_i)) \]

where \( f(x_i) = \sum_t \alpha_t h_t(x_i) \), and \( H(x_i) = \text{sign}(f(x_i)) \) is the final classifier.
Proof: Training error of AdaBoost

• \( D_1(i) = \frac{1}{m} \)

• \( D_2(i) = \frac{\exp(-y_i \alpha_1 h_1(x_i))}{mZ_1} \)

• \( D_3(i) = \frac{D_2(i) \exp(-y_i \alpha_2 h_2(x_i))}{Z_2} = \frac{\exp(-y_i \alpha_1 h_1(x_i)) \exp(-y_i \alpha_2 h_2(x_i))}{mZ_1 Z_2} \)

By induction, we have

• \( D_T(i) = \frac{D_{T-1} \exp(-y_i \alpha_T h_T(x_i))}{Z_T} = \frac{\exp(\sum_t -y_i \alpha_t h_t(x_i))}{mZ_1 Z_2 \ldots Z_T} = \frac{\exp(-y_i f(x_i))}{mZ_1 Z_2 \ldots Z_T} \)

From \( \sum_i D_T(i) = 1 \), we have \( \prod_t Z_t = \frac{1}{m} \sum_i \exp(-y_i f(x_i)) \).
Proof: Training error of AdaBoost

• So we have

\[
\frac{1}{m} \sum_i \delta(y_i \neq f(x_i)) \leq \frac{1}{m} \sum_i \exp(-y_i f(x_i)) = \Pi_t Z_t
\]

• If $Z_t < 1$, the training error decays exponentially.
Proof: Training error of AdaBoost

- Finding an optimal weight $\alpha_t$ by minimizing $Z_t$

\[
Z_t = \sum_i D_t(i) \exp(-\alpha_t y_i h_t(x_i))
\]

\[
= \sum_{h_t(x_i) \neq y_i} D_t(i) \exp(\alpha_t) + \sum_{h_t(x_i) = y_i} D_t(i) \exp(-\alpha_t)
\]

\[
= \exp(\alpha_t) \epsilon_t + \exp(-\alpha_t) (1 - \epsilon_t)
\]

\[
\frac{\partial Z_t}{\partial \alpha_t} = \exp(\alpha_t) \epsilon_t - \exp(-\alpha_t) (1 - \epsilon_t) = 0, \text{ thus}
\]

- the optimal $\alpha_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}$, and

\[
Z_t = 2 \sqrt{\epsilon_t (1 - \epsilon_t)} = \sqrt{1 - (1 - 2\epsilon_t)^2}
\]
Proof: Training error of AdaBoost

• Training error is bounded by

\[
\frac{1}{m} \sum_{i=1}^{m} \delta(H(x_i) \neq y_i) \leq \prod_{t} Z_t = \prod_{t} \sqrt{1 - (1 - 2\epsilon_t)^2} \\
\leq \exp \left( -2 \sum_{t=1}^{T} (1/2 - \epsilon_t)^2 \right)
\]

grows as $\epsilon_t$ moves away from $1/2$
Recognition Rate

- Test error still decreases even after training error reaches zero.
- Adaboost can fast decrease the training error
Comparison between LR and Boosting

• Logistic Regression assumes

\[ P(y_i = 1|x_i) = \frac{1}{1+\exp(-y_if(x_i))}, \quad f(x) = \sum_d w_d x_d + w_0 \]

• Maximizing the data log likelihood

\[
\max_w \log(1 + \exp(-y_if(x_i)))
\]

Or

\[
\min_w \log(1 + \exp(-y_if(x_i)))
\]
Comparison between LR and Boosting

• Logistic Regression (logistic loss)
  \[
  \min_{\mathbf{w}} \log(1 + \exp(-y_i f(x_i))), f(x) = \sum_d w_d x_d + w_0
  \]

• Boosting (exponential loss)
  \[
  \min_h \frac{1}{m} \sum_i \exp(-y_i f(x_i)) = \prod_t Z_t, f(x) = \sum_t \alpha_t h_t(x)
  \]

• Comparing \( \alpha_t \leftrightarrow w_d \) and \( h_t \leftrightarrow x_d \), LR and Boosting becomes comparable.
  • For LR all \( w_d \) are jointly learned, but for Boosting, \( h_t \) are learned sequentially.
  • Note that LR is linear classifier, but Boosting is not.
Difference

• A common mistake: Even if you choose a linear classifier for $h_t$, the final classifier is not linear in Boosting.
  • A sign function should be taken for $h_t(x) = \text{sign}(w^T x)$, otherwise a linear function of a set of linear functions $f(x) = \sum_t \alpha_t h_t(x)$ is still linear which we do not desire.
Summary

• We have learned to use a set of week classifiers to build a strong classifier, which
  • Can reduce the training error arbitrarily close to zero.
  • Even if the weak classifier is only slightly better than random guess

• A particular Boosting algorithm: Adaboost

• Compare the Logistic Regression and Boosting
  • Linear VS. Nonlinear
  • Joint optimization VS. iterative optimization