Stat 588 – Fall 2007
Data Mining

Lecture 9: Boosting

Improving Decision Tree Performance

- Improve accuracy through tree ensemble:
  - boosting
  - bagging
    - generate bootstrap samples.
    - train one tree per bootstrap sample.
    - take unweighted average of the trees.
  - random forest
    - bagging with additional randomization.

Ensemble Learning

- Given m classifiers $f_1, \ldots, f_m$ obtained using multiple learning algorithm.
- Ensemble is a combined classifier of the form:
  - $f(x) = \sum_{j=1}^{m} w_j f_j(x)$.
- How to build f_j and w_j simultaneously.
- Example: boosting (other methods: voting, bagging, etc).

Boosting

- Given a learning algorithm $\mathcal{A}$, how to generate ensemble?
- Invoke $\mathcal{A}$ with multiple samples (similar to Bagging).
  - goal: to find optimal ensemble by minimizing a loss function
  - learning method:
    - greedy, stage-wise optimization
    - invoking a base-learner (weak learner) $\mathcal{A}$.
    - adaptive resampling
- Bias reduction:
  - less stable but more expressive.
  - better than any single classifier.

Why boosted trees

- Build shallow trees
  - combine shallow trees (weak learner) to get strong learner.
- Linear model of high order features
  - automatically find high order interactive features
  - automatically handle heterogeneous features
  - high order features are indicator functions.
- Alternatives:
  - discretize each feature into (possibly overlapping) buckets
  - direct construction of feature combination.

- nonlinear functions like kernels or neural networks.
- direct greedy learning.

Weak learning and adaptive resampling

- $\mathcal{A}$: a weak learner (e.g. shallow tree)
  - better than chance (0.5 error) on any (reweighted) training data.
- Question: can we combine weak learners to obtain a strong learner?
- Answer: yes, through adaptive resampling (boosting).
  - idea: overweighting difficult examples that are hard to classify.
- Compare with bagging: sampling without overweighting errors.

The idea of adaptive resampling

- Reweight the training data to overweight difficult examples.
- Using weak learner $\mathcal{A}$ to obtain classifiers f_j on reweighted samples.
- Adding the new classifier into ensemble, and choose weight w_j.
- Iterate.
- Final classifier is $\sum_j w_j f_j$.

AdaBoost (adaptive boosting)

- How to reweight, and how to compute w.
- Assume binary classification $y \in \{\pm 1\}$, and $f \in \{\pm 1\}$.

Table 1: AdaBoost

initialize sample weights $\{d_i\} = \{1/n\}$ for $\{(X_i, Y_i)\}$
for $j = 1, \cdots, J$
call *Weak Learner* to obtain f_j using sample weighted by $\{d_i\}$
let $r_j = \sum_i d_i f_j(X_i) Y_i$
let $w_j = 0.5 \ln((1 + r_j)/(1 - r_j))$
update d_i: $d_i \propto d_i e^{-w_j f_j(X_i) Y_i}$.
let $\bar{f}_J(x) = \sum_{j=1}^{J} w_j f_j(x)$

Some theoretical results about AdaBoost

- Convergence:
  - reduces margin error:
    - f correctly classifies X_i with margin γ if $\boldsymbol{f(X_i)Y_i > \gamma > 0}$.
    - If each weak learner f_j does better than $0.5 - \delta_j$ ($\delta_j > 0$) on reweighted samples with respect to classification error $I(f(X_i)Y_i \leq 0)$, then

$$\frac{1}{n}\sum_{i=1}^{n} I(\bar{f}_J(X_i)Y_i \leq \gamma) \leq \exp(\gamma - 2\sum_{j=1}^{J}\delta_j^2).$$

- Generalization: large margin implies good generalization performance.
  - for separable problems, Adaboost does not usually maximize margin.

Generalization analysis

- Generalization performance of $\hat{f} = \mathcal{A}(S_n)$: with probability at least $1 - \eta$,

$$\text{test error} \leq \text{training error} + \text{model complexity}.$$

- Decision tree of fixed depth: $\mathcal{H}$ has finite VC-dimension d_{VC}, ($\phi(f, y) = I(fy \leq 0)$):

$$\text{test error} \leq \text{training error} + C\sqrt{\frac{1}{n}(d_{VC} - \ln(\eta))}$$

$$\text{test error} \leq 2 \times \text{training error} + \frac{C}{n}(d_{VC} - \ln(\eta)).$$

Generalization error of boosting using number of steps

- $\mathcal{H}$: VC-dimension d_{VC}.
- Ensemble $\bar{f}_J = \sum_{i=1}^{J} w_i f_i(x) : f_i \in \mathcal{H}\}$:

$$\text{test error} \leq 2 \times \text{training error} + \underbrace{\frac{C}{n}(Jd_{VC} - \ln(\eta))}_{\text{complexity linear in } J}.$$

- $\bar{f}_J$: boosted tree after J round:
  - training error: $O(e^{-2J\delta^2})$ ($0.5 - \delta$ error reduction)
  - generalization error

$$R(\bar{f}_J) \leq O(e^{-J\gamma}) + \frac{C}{n}(Jd_{VC} - \ln(\eta)).$$

Generalization error anomaly

- Observations:
  - AdaBoost is difficult to overfit.
  - even when training error becomes zero, generalization error still decays
- Not explained by the generalization bound using the number of steps.
- require additional analysis: margin

Margin bound

- Decision tree of fixed depth: $\mathcal{H}$ has finite VC-dimension d_{VC}, then

$$\text{training error} \leq 2 \times \text{margin error} + \text{complexity}$$

$$\mathbf{E}_{X,Y} I(\bar{f}_J(X)Y \leq 0) \leq \underbrace{\frac{2}{n}\sum_{i=1}^{n} I(\hat{f}_m(X_i)Y_i \leq \gamma \sum_{j=1}^{J} w_j)}_{\to 0 \text{ when } J \to \infty} + \underbrace{\frac{C}{n}(\gamma^{-2} d_{VC} - \ln(\eta))}_{\text{independent of } J}.$$

- Explains why AdaBoost can keep improving even when classification error becomes zero
  - margin error decreases

Margin analysis and 1-norm regularization

- Margin analysis is a special case of general 1-norm regularization
- Let ϕ be a smooth loss.
- Given 1-norm constraint $\sum_j w_j \leq A$:

$$\mathbf{E}_{X,Y}\phi(\bar{f}_J(X), Y) \leq \frac{1}{n}\sum_{i=1}^{n}\phi(\bar{f}_J(X_i), Y_i) + C_\phi\sqrt{\frac{1}{n}(A^2 d_{VC} - \ln(\eta))}.$$

Complexity measured by A, not number of steps J.

Summary of Generalization Analysis

- Estimate generalization of boosting: using the following complexity control
  - L_1: 1-norm of the weights w_j are bounded.
  - L_0: number of boosting steps (sparse representation).
- Which complexity control is better?
  - sparsity is more fundamental but both views are useful.
  - can be more refined analysis in between.
- In more general boosting methods:
  - complexity can be controlled either by L_1 (1-norm) or L_0 (sparsity).

Issues corresponding to the weak learner view

- Weak learner: this is only an assumption, how can we prove it exists.
  - what is a weak learner anyway: why boosted tree works, and boosted SVM does not.
- Overfitting: driving error to zero can overfit the data (for non-separable problems)
- AdaBoost does not maximize margin.
- Adaptive resampling: why this specific form.
- Can we generalize adaptive resampling idea to regression and complex loss functions?

From adaptive resampling to greedy boosting

- Weak learner: picks f_j from a hypothesis space $\mathcal{H}_j$ to minimize certain error criterion.

- Goal: find $w_j \geq 0$ and $f_j \in \mathcal{H}_j$ to minimize loss

$$[\{\hat{w}_j, \hat{f}_j\}] = \arg \min_{\{w_j \geq 0, f_j \in \mathcal{H}_j\}} \sum_{i=1}^{n} \phi \left(\sum_j w_j f_j(X_i), Y_i \right). \qquad (*)$$

- Idea: greedy optimization.
  - at stage j: fix (w_k, f_k) ($k < j$), find (w_j, f_j) to minimize the loss $(*)$.

AdaBoost as greedy boosting

- Loss $\phi(f, y) = \exp(-fy)$.
- Goal: using greedy boosting to minimize

$$[\{\hat{w}_j, \hat{f}_j\}] = \arg \min_{\{w_j \geq 0, f_j \in \mathcal{H}_j\}} \sum_{i=1}^{n} e^{-\sum_j w_j f_j(X_i) Y_i}.$$

- At stage j, let $d_i \propto e^{-\sum_{k<j} \hat{w}_k \hat{f}_k(X_i) Y_i}$, and solve

$$[\hat{w}_j, \hat{f}_j] = \arg \min_{w_j \geq 0, f_j \in \mathcal{H}_j} \sum_{i=1}^{n} d_i e^{-w_j f_j(X_i) Y_i}.$$

- Let $\bar{f}(x) = \sum_k \hat{w}_k \hat{f}_k(x)$.
- Solution of $\hat{w}_j$ with fixed $\hat{f}_j$:

  $D_{-1}(\hat{f}_j) = (1 - r_j)/2 = \sum_{i:\hat{f}_j(X_i)Y_i=-1} d_i$ (classification error):

$$\hat{w}_j = 0.5 \ln((1 - D_{-1})/D_{-1}), \quad \sum_{i=1}^{n} d_i e^{-\hat{w}_j \hat{f}_j(X_i) Y_i} = 2\sqrt{(1 - D_{-1})D_{-1}}$$

- Optimal $\hat{f}$: classifier minimizing error with reweighted samples d_i.
- Stage-wise exponential loss minimization (AdaBoost procedure):
  - choose $\hat{f}_j \in \mathcal{H}_j$ to minimize classification error
  - let $\hat{w}_j = 0.5 \ln((1 - D_{-1})/D_{-1})$
  - exactly leads to the AdaBoost procedure.

General Loss Function

- Learn prediction function $\boldsymbol{h(x)}$: input x and output y
- By solving learning formulation

$$\hat{h} = \arg\min_{h \in H} R(h)$$

  – $R(h)$: complex loss function of the form

$$h = \frac{1}{n} \sum_{i=1}^{n} \phi_i(h(x_{i,1}), \cdots, h(x_{i,m_i}), y_i)$$

- Greedy algorithm: generalization of Adaboost

- $(s_k, g_k) = \arg\min_{g \in C, s \in R} R(h_k + sg)$
- $h_{k+1} \leftarrow h_k + \tilde{s}_k g_k$ ($\tilde{s}_k$ may not equal s_k)

Why boosted tree works

- Linear model of high order features
- Automatically handle heterogeneous features
  - create new (high order) features that are indicator functions.
- Automatically find high order interactive features
  - through tree splitting procedure.
  - a method to solve the problem of huge search space.
    - assume good high order features depend on actively maintained set of (good) features constructed so far.
- Alternatives:

- discretize each feature into (possibly overlapping) buckets
- direct construction of feature combination.
- nonlinear functions like kernels or neural networks.
- general greedy feature learning by maintaining a set of features and adding new ones.

Greedy Boosting in Convex Hull

Solving the optimization problem: $\inf_{f \in \text{co}(S)} A(f)$, where $\text{co}(S)$ is the convex hull of S.

The algorithm:

- Start with $f_0 \in S$.
- **for** $k = 1, 2, \ldots$
  - Find $\bar{g}_k \in S$ and $0 \le \bar{\alpha}_k \le 1$ to approximately minimize the function:
    $(\alpha_k, g_k) \to A((1-\alpha_k) f_{k-1} + \alpha_k g_k) \qquad (*)$
  - Let $f_k = (1 - \bar{\alpha}_k) f_{k-1} + \bar{\alpha}_k \bar{g}_k$.

$(*)$ step (weak-learning): $A((1-\bar{\alpha}_k) f_{k-1} + \bar{\alpha}_k \bar{g}_k) \le \inf_{g,\alpha} A((1-\alpha) f_{k-1} + \alpha g)$.

One-step analysis

- Goal: obtain upper bound of

$$A^+(v) = \inf_{\eta \in [0,1], u \in S} A((1-\eta)v + \eta u).$$

- Averaging technique:
  - Consider an arbitrary $w = \sum_{i=1}^m \alpha_i u_i \in \text{co}(S)$
    * $\alpha_i \geq 0$ and $\sum_{i=1}^m \alpha_i = 1$, $u_i \in S$.
  - Design the following rule parameterized by $\eta \in [0,1]$:

$$B(\eta) = \sum_{i=1}^m \alpha_i A((1-\eta)v + \eta u_i).$$

- Observe: $A^+(v) \leq \inf_{v,\eta} B(\eta)$

Intuition

Given $w = \sum_{i=1}^{m} \alpha_i u_i \in \mathrm{co}(S)$. First order approximation:

$$
\begin{aligned}
A^+(v) \leq & B(\eta) \\
= & \sum_{i=1}^{m} \alpha_i A((1-\eta)v + \eta u_i) \\
= & \sum_{i=1}^{m} \alpha_i [A(v) - \eta \nabla A(v)^T (v - u_i)] + O(\eta^2) \\
= & A(v) - \eta \nabla A(v)^T (v - w) + O(\eta^2) \\
= & A(v) - \eta (A(v) - A(w)) + O(\eta^2).
\end{aligned}
$$

Some Observations

- Minimize r.h.s over $w \in \mathsf{CO}(S)$:

$$A^{+}(v) \leq A(v) - \eta(A(v) - \inf_{w \in \mathsf{CO}(S)} A(w)) + O(\eta^2).$$

More precise derivation

- Some technical tools
  - Convexity property: $A(w) - A(v) - \nabla A(v)^T (w - v) \geq 0$.
  - Taylor expansion: $A((1-\eta)v + \eta v') - A(v) \leq \eta (v' - v)^T \nabla A(v) + \frac{\eta^2}{2} M$.
- Assumption: $M = \sup_{v \in \mathrm{CO}(S), u \in S, \theta \in (0,1)} \frac{d^2}{d\theta^2} A(v + \theta(u - v)) < +\infty$.

$$
\begin{aligned}
B(\eta) \leq & A(v) - \eta \nabla A(v)^T (v - w) + \frac{\eta^2}{2} M \\
\leq & A(v) - \eta (A(v) - A(w)) + \frac{\eta^2}{2} M.
\end{aligned}
$$

Optimize the one-step convergence bound

$\forall w \in \text{CO}(S)$ and $\eta \in [0,1]$:

$$A(f_{k+1}) - A(w) \leq A(f_k) - A(w) - \eta(A(f_k) - A(w)) + \frac{\eta^2}{2}M.$$

Let $A(w) \to \inf_{w \in \text{CO}(S)} A(w)$, and define

$$\rho(v) = A(v) - \inf_{w \in \text{CO}(S)} A(w).$$

Optimize over $\eta \in [0,1]$:

$$\boldsymbol{\rho(f_{k+1}) \leq \begin{cases} \rho(f_k) - \frac{\rho(f_k)^2}{2M} & \textbf{if } \rho(f_k) \leq M, \\ \frac{M}{2} & \textbf{otherwise.} \end{cases}}$$

Convergence rate

- Recursion of $b(k) = \rho(f_k)$: $\boldsymbol{b(k+1) \leq b(k) - b(k)^2/(2M)}$.
- Asymptotic expression:
  - $b'(k) \approx -b(k)^2/(2M)$
  - $1/b(k) \approx k/(2M) + c_0$
- The solution:
  - Plug-in the asymptotic form, and use induction.
  - After one-step: $A(f_1) \leq M/2$.
  - After $k \geq 1$ step:
  $$A(f_k) \leq 2M/(k+3).$$

References

- AdaBoost

  Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

- Convex hull boosting analysis:

  T. Zhang. Sequential greedy approximation for certain convex optimization problems. *IEEE Transaction on Information Theory*, 49:682–691, 2003.

- Greedy boosting:

  T. Zhang and B. Yu. Boosting with early stopping: Convergence and consistency. *The Annals of Statistics*, 33:1538–1579, 2005.