Deep Boosting

Joint work with Corinna Cortes (Google Research)
Umar Syed (Google Research)
Ensemble Methods in ML

- Combining several base classifiers to create a more accurate one.
  - Bagging (Breiman 1996).
  - AdaBoost (Freund and Schapire 1997).
  - Stacking (Smyth and Wolpert 1999).
  - Bayesian averaging (MacKay 1996).
  - Other averaging schemes e.g., (Freund et al. 2004).

- Often very effective in practice.
- Benefit of favorable learning guarantees.
Convex Combinations

- Base classifier set $H$.
  - boosting stumps.
  - decision trees with limited depth or number of leaves.
- Ensemble combinations: convex hull of base classifier set.

$$
\text{conv}(H) = \left\{ \sum_{t=1}^{T} \alpha_t h_t : \alpha_t \geq 0; \sum_{t=1}^{T} \alpha_t \leq 1; \forall t, h_t \in H \right\}.
$$
Rademacher Complexity

 Definitions: let $H$ be a family of functions mapping from $X$ to $\mathbb{R}$ and let $S = (x_1, \ldots, x_m)$ be a sample of size $m$.

 Empirical Rademacher complexity of $H$:

$$\hat{\mathcal{R}}_S(H) = \frac{1}{m} \mathbb{E}_\sigma \left[ \sup_{h \in H} \sum_{i=1}^{m} \sigma_i h(x_i) \right].$$

 Rademacher complexity of $H$:

$$\mathcal{R}_m(H) = \mathbb{E}_{S \sim D^m} \left[ \hat{\mathcal{R}}_S(H) \right].$$
Ensembles - Margin Bound

(Koltchinskii and Panchenko, 2002)

Theorem: let $H$ be a family of real-valued functions. Fix $\rho > 0$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds for all $f = \sum_{t=1}^{T} \alpha_t h_t \in \text{conv}(H)$:

$$R(f) \leq \hat{R}_{S,\rho}(f) + \frac{2}{\rho} \mathcal{R}_m(H) + \sqrt{\frac{\log \frac{1}{\delta}}{2m}},$$

where $\hat{R}_{S,\rho}(f) = \frac{1}{m} \sum_{i=1}^{m} 1_{y_i f(x_i) \leq \rho}$.
Questions

- Can we use a much richer or deeper base classifier set?
  - richer families needed for difficult tasks on speech and image processing.
  - but generalization bound indicates risk of overfitting.
AdaBoost

(Freund and Schapire, 1997)

- **Description:** coordinate descent applied to

\[ F(\alpha) = \sum_{i=1}^{m} e^{-y_i f(x_i)} = \sum_{i=1}^{m} \exp \left( -y_i \sum_{t=1}^{T} \alpha_t h_t(x_i) \right). \]

- **Guarantees:** ensemble margin bound.

  - but AdaBoost does not maximize the margin!
  - some margin maximizing algorithms such as arc-gv are outperformed by AdaBoost! (Reyzin and Schapire, 2006)
Suspicions

- Complexity of hypotheses used:
  - arc-gv tends to use deeper decision trees to achieve a larger margin.

- Notion of margin:
  - minimal margin perhaps not the appropriate notion.
  - margin distribution is key.

  can we shed more light on these questions?
This Talk

**Main question:** how can we design ensemble algorithms that can succeed even with very deep decision trees or other complex sets?

- theory.
- practical algorithms.
- experimental results.
Theory
Base Classifier Set $\mathcal{H}$

- Decomposition in terms of sub-families or their union.
Non-negative linear ensembles $\mathcal{F} = \text{conv}(\bigcup_{k=1}^{p} H_k)$:

$$f = \sum_{t=1}^{T} \alpha_t h_t$$

with $\alpha_t \geq 0$, $\sum_{t=1}^{T} \alpha_t \leq 1$, $h_t \in H_{k_t}$. 
Ideas

- Use hypotheses drawn from $H_k s$ with larger $k$'s but allocate more weight to hypotheses drawn from smaller $k$'s.
  - how can we determine quantitatively the amounts of mixture weights apportioned to different families?
  - can we provide learning guarantees guiding these choices?
  - connection with SRM?
New Learning Guarantee

Theorem: Fix $\rho > 0$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds for all $f = \sum_{t=1}^{T} \alpha_t h_t \in \mathcal{F}$:

$$R(f) \leq \hat{R}_{S,\rho}(f) + \frac{4}{\rho} \sum_{t=1}^{T} \alpha_t \mathcal{K}_m(H_{k_t}) + \tilde{O}\left(\sqrt{\frac{\log p}{\rho^2 m}}\right).$$
Consequences

- Complexity term with explicit dependency on mixture weights.
  - quantitative guide for controlling weights assigned to more complex sub-families.
  - bound can be used to inspire, or directly define an ensemble algorithm.
Algorithms
Set-Up

- $H_1, \ldots, H_p$: disjoint sub-families of functions taking values in $[-1, +1]$.

- Further assumption (not necessary): symmetric sub-families, i.e. $h \in H_k \iff -h \in H_k$.

- Notation: for any $h \in \bigcup_{k=1}^{p} H_k$,
  - $d(h)$ index of sub-family ($h \in H_{d(h)}$).
  - $r_t = \mathcal{R}_m(H_{d(h_t)})$. 
Derivation (1)

Learning bound suggests seeking $\alpha \geq 0$ with $\sum_{t=1}^{T} \alpha_t \leq 1$ to minimize

$$
\frac{1}{m} \sum_{i=1}^{m} y_i \sum_{t=1}^{T} \alpha_t h_t(x_i) \leq \rho + \frac{4}{\rho} \sum_{t=1}^{T} \alpha_t r_t.
$$
Derivation (2)

Since $f = \sum_{t=1}^{T} \alpha_t h_t$ and $f / \rho$ have the same error, we can instead search for $\alpha \geq 0$ with $\sum_{t=1}^{T} \alpha_t \leq \frac{1}{\rho}$ to minimize

$$
\frac{1}{m} \sum_{i=1}^{m} 1_{y_i \sum_{t=1}^{T} \alpha_t h_t(x_i) \leq 1} + 4 \sum_{t=1}^{T} \alpha_t r_t.
$$
Derivation (3)

- Let \( u \mapsto \Phi(-u) \) be a decreasing convex function upper bounding \( u \mapsto 1_{u \leq 0} \), with \( \Phi \) differentiable.

- Minimizing convex upper bound (convex optimization):

\[
\min_{\alpha \geq 0} \frac{1}{m} \sum_{i=1}^{m} \Phi \left( 1 - y_i \sum_{t=1}^{T} \alpha_t h_t(x_i) \right) + \lambda \sum_{t=1}^{T} \alpha_t r_t \\
\text{s.t.} \quad \sum_{t=1}^{T} \alpha_t \leq \frac{1}{\rho},
\]

where the parameter \( \lambda \geq 0 \) is introduced.
Moving the constraint to the objective and using the fact that the sub-families are symmetric leads to:

\[
\min_{\alpha \in \mathbb{R}^N} \frac{1}{m} \sum_{i=1}^{m} \Phi \left( 1 - y_i \sum_{j=1}^{N} \alpha_j h_j(x_i) \right) + \sum_{t=1}^{N} (\lambda r_j + \beta) |\alpha_j|,
\]

where \( \lambda, \beta \geq 0 \), and for each hypothesis, keep either \( h \) or \( -h \).
Convex Surrogates

Two principal choices:

- Exponential loss: $\Phi(-u) = \exp(-u)$.
- Logistic loss: $\Phi(-u) = \log_2(1 + \exp(-u))$.
DeepBoost Algorithm

- Coordinate descent applied to convex objective.
  - non-differentiable function.
  - definition of maximum coordinate descent.
Direction & Step

- Maximum direction: definition based on the error
  \[
  \epsilon_{t,j} = \frac{1}{2} \left[ 1 - \mathbb{E}_{i \sim D_t} [y_i h_j(x_i)] \right],
  \]
  where $D_t$ is the distribution over sample at iteration $t$.

- Step:
  - closed-form expressions for exponential and logistic losses.
  - general case: line search.
Pseudocode

\[
\text{DEEPBOOST}(S = ((x_1, y_1), \ldots, (x_m, y_m)))
\]

1. for \( i \leftarrow 1 \) to \( m \) do
2. \( D_1(i) \leftarrow \frac{1}{m} \)
3. for \( t \leftarrow 1 \) to \( T \) do
4. for \( j \leftarrow 1 \) to \( N \) do
5. \quad if \( (\alpha_{t-1,j} \neq 0) \) then
6. \quad \quad \( d_j \leftarrow (\epsilon_{t,j} - \frac{1}{2}) + \text{sgn}(\alpha_{t-1,j}) \frac{\Lambda_{j,m}}{2S_t} \)
7. \quad \quad else if \( |\epsilon_{t,j} - \frac{1}{2}| \leq \frac{\Lambda_{j,m}}{2S_t} \) then
8. \quad \quad \quad \( d_j \leftarrow 0 \)
9. \quad \quad else \( d_j \leftarrow (\epsilon_{t,j} - \frac{1}{2}) - \text{sgn}(\epsilon_{t,j} - \frac{1}{2}) \frac{\Lambda_{j,m}}{2S_t} \)
10. \quad \quad \( k \leftarrow \arg\max_{j \in [1,N]} |d_j| \)
11. \( \epsilon_t \leftarrow \epsilon_t,k \)
12. \quad if \( (1 - \epsilon_t)e^{\alpha_{t-1,k}} - \epsilon_t e^{-\alpha_{t-1,k}} \leq \frac{\Lambda_{k,m}}{S_t} \) then
13. \quad \quad \( \eta_t \leftarrow -\alpha_{t-1,k} \)
14. \quad \quad else if \( (1 - \epsilon_t)e^{\alpha_{t-1,k}} - \epsilon_t e^{-\alpha_{t-1,k}} > \frac{\Lambda_{k,m}}{S_t} \) then
15. \quad \quad \quad \( \eta_t \leftarrow \log \left[ -\frac{\Lambda_{k,m}}{2\epsilon_t S_t} + \sqrt{\left( \frac{\Lambda_{k,m}}{2\epsilon_t S_t} \right)^2 + \frac{1 - \epsilon_t}{\epsilon_t}} \right] \)
16. \quad \quad \quad else \( \eta_t \leftarrow \log \left[ +\frac{\Lambda_{k,m}}{2\epsilon_t S_t} + \sqrt{\left( \frac{\Lambda_{k,m}}{2\epsilon_t S_t} \right)^2 + \frac{1 - \epsilon_t}{\epsilon_t}} \right] \)
17. \quad \( \alpha_t \leftarrow \alpha_{t-1} + \eta_t e_k \)
18. \quad \( S_{t+1} \leftarrow \sum_{i=1}^{m} \Phi'(1 - y_i \sum_{j=1}^{N} \alpha_{t,j} h_j(x_i)) \)
19. for \( i \leftarrow 1 \) to \( m \) do
20. \quad \( D_{t+1}(i) \leftarrow \Phi'(1 - y_i \sum_{j=1}^{N} \alpha_{t,j} h_j(x_i))/S_{t+1} \)
21. \( f \leftarrow \sum_{j=1}^{N} \alpha_{t,j} h_j \)
22. return \( f \)
Notes

- Straightforward updates.
- Sparsity step.
- Parallel implementation.
Connections with Previous Work

For $\lambda = \beta = 0$, DeepBoost coincides with
- AdaBoost (Freund and Schapire 1997), run with union of sub-families, for the exponential loss.
- algorithm of (Friedman et al., 1998), run with union of sub-families, for the logistic loss.

For $\lambda = 0$ and $\beta \neq 0$, DeepBoost
- coincides with L1-regularized AdaBoost (Duchi and Singer 2009).
- close to unnormalized Arcing (Breiman 1999).
- also related to AdaBoost$_p$ (Raetsch and Warmuth 2005).
Experiments
Rad. Complexity Estimates

- Benefit of data-dependent analysis:
  - empirical estimates of each $\mathcal{R}_m(H_k)$.
  - example: for kernel function $K_k$,
    \[
    \hat{\mathcal{R}}_S(H_k) \leq \frac{\sqrt{\text{Tr}[K_k]}}{m}.
    \]
  - alternatively, upper bounds in terms of growth functions,
    \[
    \mathcal{R}_m(H_k) \leq \frac{\sqrt{2 \log \Pi_{H_k}(m)}}{m}.
    \]
Experiments (1)

- Family of base classifiers defined by boosting stumps:
  - boosting stumps $H_1^\text{stumps}$ (threshold functions).
  - in dimension $d$, $\prod_{H_1^\text{stumps}}(m) \leq 2md$, thus
    $$\mathcal{R}_m(H_1^\text{stumps}) \leq \sqrt{\frac{2 \log(2md)}{m}}.$$  
  - decision trees of depth 2, $H_2^\text{stumps}$, with the same question at the internal nodes of depth 1.
    - in dimension $d$, $\prod_{H_2^\text{stumps}}(m) \leq (2m)^2 \frac{d(d-1)}{2}$, thus
      $$\mathcal{R}_m(H_2^\text{stumps}) \leq \sqrt{\frac{2 \log(2m^2d(d - 1))}{m}}.$$
Experiments (1)

- Base classifier set: $H^\text{stumps}_1 \cup H^\text{stumps}_2$.

- Data sets:
  - same UCI Irvine data sets as (Breiman 1999) and (Reyzin and Schapire 2006).
  - OCR data sets used by (Reyzin and Schapire 2006): ocr17, ocr49.
  - MNIST data sets: ocr17-mnist, ocr49-mnist.

- Experiments with exponential loss.

Data Statistics

<table>
<thead>
<tr>
<th></th>
<th>breastcancer</th>
<th>ionosphere</th>
<th>german (numeric)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>699</td>
<td>351</td>
<td>1000</td>
</tr>
<tr>
<td>Attributes</td>
<td>9</td>
<td>34</td>
<td>24</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>diabetes</th>
<th>ocr17</th>
<th>ocr49</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>768</td>
<td>2000</td>
<td>2000</td>
</tr>
<tr>
<td>Attributes</td>
<td>8</td>
<td>196</td>
<td>196</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>ocr17-mnist</th>
<th>ocr49-mnist</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>15170</td>
<td>13782</td>
</tr>
<tr>
<td>Attributes</td>
<td>400</td>
<td>400</td>
</tr>
</tbody>
</table>
Experiments - Stumps Exp Loss

Table 1. Results for boosted decision stumps and the exponential loss function.

<table>
<thead>
<tr>
<th></th>
<th>breastcancer</th>
<th>ionosphere</th>
<th>german</th>
<th>diabetes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error</td>
<td>0.0429</td>
<td>0.1014</td>
<td>0.243</td>
<td>0.253</td>
</tr>
<tr>
<td>(std dev)</td>
<td>(0.0248)</td>
<td>(0.0414)</td>
<td>(0.0445)</td>
<td>(0.0330)</td>
</tr>
<tr>
<td>Avg tree size</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Avg no. of trees</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>AdaBoost</td>
<td>0.0437</td>
<td>0.075</td>
<td>0.2505</td>
<td>0.260</td>
</tr>
<tr>
<td>$H_2^{stamps}$</td>
<td>(0.0214)</td>
<td>(0.0413)</td>
<td>(0.0487)</td>
<td>(0.0518)</td>
</tr>
<tr>
<td>AdaBoost-L1</td>
<td>0.0408</td>
<td>0.0708</td>
<td>0.2455</td>
<td>0.254</td>
</tr>
<tr>
<td>$H_2^{stamps}$</td>
<td>(0.0223)</td>
<td>(0.0331)</td>
<td>(0.0438)</td>
<td>(0.0486)</td>
</tr>
<tr>
<td>DeepBoost</td>
<td>0.0373</td>
<td>0.0638</td>
<td>0.2395</td>
<td>0.253</td>
</tr>
<tr>
<td></td>
<td>(0.00225)</td>
<td>(0.00394)</td>
<td>(0.00462)</td>
<td>(0.00510)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>ocr17</th>
<th>ocr49</th>
<th>ocr17-mnist</th>
<th>ocr49-mnist</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error</td>
<td>0.0085</td>
<td>0.0555</td>
<td>0.0056</td>
<td>0.0414</td>
</tr>
<tr>
<td>(std dev)</td>
<td>0.0072</td>
<td>0.0414</td>
<td>0.0017</td>
<td>0.00353</td>
</tr>
<tr>
<td>Avg tree size</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Avg no. of trees</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>AdaBoost</td>
<td>0.0085</td>
<td>0.032</td>
<td>0.0048</td>
<td>0.0209</td>
</tr>
<tr>
<td>$H_1^{stamps}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdaBoost-L1</td>
<td>0.0075</td>
<td>0.0122</td>
<td>0.0046</td>
<td>0.0200</td>
</tr>
<tr>
<td>$H_1^{stamps}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DeepBoost</td>
<td>0.0070</td>
<td>0.0068</td>
<td>0.0044</td>
<td>0.0177</td>
</tr>
<tr>
<td></td>
<td>(0.0048)</td>
<td></td>
<td>(0.00438)</td>
<td></td>
</tr>
</tbody>
</table>
Family of base classifiers defined by decision trees of depth $k$.

Since $\text{VC-dim}(H^\text{trees}_k) \leq (2^k + 1) \log_2 (d + 1)$,

$$
\mathcal{R}_m(H^\text{trees}_k) \leq \sqrt{\frac{(2^{k+1} + 2) \log_2 (d + 1) \log(m)}{m}}.
$$

Base classifier set: $\bigcup_{k=1}^{K} H^\text{trees}_k$.

Same data sets as with Experiments (1).

Both exponential and logistic loss.

Comparison with AdaBoost and AdaBoost-L1.
### Experiments - Trees Exp Loss

*Table 2.* Results for boosted decision trees and the exponential loss function.

<table>
<thead>
<tr>
<th></th>
<th>breastcancer</th>
<th></th>
<th></th>
<th></th>
<th>ocr17</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>ocr49</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>ocr17-mnist</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>ocr49-mnist</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>0.0267</td>
<td>0.0264</td>
<td><strong>0.0243</strong></td>
<td></td>
<td>0.004</td>
<td>0.003</td>
<td><strong>0.002</strong></td>
<td></td>
<td>0.0180</td>
<td></td>
<td>0.0175</td>
<td></td>
<td><strong>0.0175</strong></td>
<td></td>
<td>0.00471</td>
<td></td>
<td>0.00471</td>
<td></td>
<td><strong>0.00409</strong></td>
<td></td>
</tr>
<tr>
<td>(std dev)</td>
<td>(0.00841)</td>
<td>(0.0098)</td>
<td><strong>(0.00797)</strong></td>
<td></td>
<td>(0.00316)</td>
<td>(0.00100)</td>
<td><strong>(0.00100)</strong></td>
<td></td>
<td>(0.00555)</td>
<td></td>
<td>(0.00357)</td>
<td></td>
<td><strong>(0.00510)</strong></td>
<td></td>
<td>(0.0022)</td>
<td></td>
<td>(0.0021)</td>
<td></td>
<td><strong>(0.0021)</strong></td>
<td></td>
</tr>
<tr>
<td>Avg tree size</td>
<td>29.1</td>
<td>28.9</td>
<td>20.9</td>
<td></td>
<td>15.0</td>
<td>30.4</td>
<td>26.0</td>
<td></td>
<td>30.9</td>
<td></td>
<td>62.1</td>
<td></td>
<td>30.2</td>
<td></td>
<td>15</td>
<td></td>
<td>66.8</td>
<td></td>
<td>59.2</td>
<td></td>
</tr>
<tr>
<td>Avg no. of trees</td>
<td>100</td>
<td>51.7</td>
<td>55.9</td>
<td></td>
<td>100</td>
<td>65.3</td>
<td>61.8</td>
<td></td>
<td>100</td>
<td></td>
<td>89.0</td>
<td></td>
<td>83.0</td>
<td></td>
<td>100</td>
<td></td>
<td>81.1</td>
<td></td>
<td>80.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ionosphere</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error</td>
<td>0.0661</td>
<td>0.0657</td>
<td><strong>0.0501</strong></td>
<td></td>
<td>0.0180</td>
<td>0.0175</td>
<td><strong>0.0175</strong></td>
<td></td>
<td>0.239</td>
<td></td>
<td>0.239</td>
<td></td>
<td><strong>0.234</strong></td>
<td></td>
<td>0.00471</td>
<td></td>
<td>0.00471</td>
<td></td>
<td><strong>0.00409</strong></td>
<td></td>
</tr>
<tr>
<td>(std dev)</td>
<td>(0.0315)</td>
<td>(0.0257)</td>
<td><strong>(0.0316)</strong></td>
<td></td>
<td>(0.00555)</td>
<td>(0.00357)</td>
<td><strong>(0.00510)</strong></td>
<td></td>
<td>(0.0165)</td>
<td></td>
<td>(0.0201)</td>
<td></td>
<td>(0.0148)</td>
<td></td>
<td>(0.0022)</td>
<td></td>
<td>(0.0021)</td>
<td></td>
<td>(0.0021)</td>
<td></td>
</tr>
<tr>
<td>Avg tree size</td>
<td>29.8</td>
<td>31.4</td>
<td>26.1</td>
<td></td>
<td>30.9</td>
<td>62.1</td>
<td>30.2</td>
<td></td>
<td>3</td>
<td></td>
<td>7</td>
<td></td>
<td>16.0</td>
<td></td>
<td>15</td>
<td></td>
<td>66.8</td>
<td></td>
<td>59.2</td>
<td></td>
</tr>
<tr>
<td>Avg no. of trees</td>
<td>100</td>
<td>69.4</td>
<td>50.0</td>
<td></td>
<td>100</td>
<td>89.0</td>
<td>83.0</td>
<td></td>
<td>100</td>
<td></td>
<td>89.0</td>
<td></td>
<td>83.0</td>
<td></td>
<td>100</td>
<td></td>
<td>81.1</td>
<td></td>
<td>80.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>german</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error</td>
<td>0.239</td>
<td>0.239</td>
<td><strong>0.234</strong></td>
<td></td>
<td>0.00471</td>
<td>0.00471</td>
<td><strong>0.00409</strong></td>
<td></td>
<td>0.249</td>
<td></td>
<td>0.240</td>
<td></td>
<td><strong>0.230</strong></td>
<td></td>
<td>0.00471</td>
<td></td>
<td>0.00471</td>
<td></td>
<td><strong>0.00409</strong></td>
<td></td>
</tr>
<tr>
<td>(std dev)</td>
<td>(0.0165)</td>
<td>(0.0201)</td>
<td><strong>(0.0148)</strong></td>
<td></td>
<td>(0.0022)</td>
<td>(0.0021)</td>
<td><strong>(0.0021)</strong></td>
<td></td>
<td>(0.0272)</td>
<td></td>
<td>(0.0313)</td>
<td></td>
<td>(0.0399)</td>
<td></td>
<td>(0.00500)</td>
<td></td>
<td>(0.00512)</td>
<td></td>
<td>(0.00551)</td>
<td></td>
</tr>
<tr>
<td>Avg tree size</td>
<td>3</td>
<td>7</td>
<td>16.0</td>
<td></td>
<td>15</td>
<td>33.4</td>
<td>22.1</td>
<td></td>
<td>29.9</td>
<td></td>
<td>66.3</td>
<td></td>
<td>30.1</td>
<td></td>
<td>29.9</td>
<td></td>
<td>81.1</td>
<td></td>
<td>80.9</td>
<td></td>
</tr>
<tr>
<td>Avg no. of trees</td>
<td>100</td>
<td>87.5</td>
<td>14.1</td>
<td></td>
<td>100</td>
<td>66.8</td>
<td>59.2</td>
<td></td>
<td>100</td>
<td></td>
<td>81.1</td>
<td></td>
<td>80.9</td>
<td></td>
<td>100</td>
<td></td>
<td>81.1</td>
<td></td>
<td>80.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>diabetes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error</td>
<td>0.249</td>
<td>0.240</td>
<td><strong>0.230</strong></td>
<td></td>
<td>0.00471</td>
<td>0.00471</td>
<td><strong>0.00409</strong></td>
<td></td>
<td>0.249</td>
<td></td>
<td>0.240</td>
<td></td>
<td><strong>0.230</strong></td>
<td></td>
<td>0.00471</td>
<td></td>
<td>0.00471</td>
<td></td>
<td><strong>0.00409</strong></td>
<td></td>
</tr>
<tr>
<td>(std dev)</td>
<td>(0.0272)</td>
<td>(0.0313)</td>
<td><strong>(0.0399)</strong></td>
<td></td>
<td>(0.00500)</td>
<td>(0.00512)</td>
<td><strong>(0.00551)</strong></td>
<td></td>
<td>(0.0272)</td>
<td></td>
<td>(0.0313)</td>
<td></td>
<td>(5.37)</td>
<td></td>
<td>(0.00500)</td>
<td></td>
<td>(0.00512)</td>
<td></td>
<td>(0.00551)</td>
<td></td>
</tr>
<tr>
<td>Avg tree size</td>
<td>3</td>
<td>3</td>
<td>5.37</td>
<td></td>
<td>29.9</td>
<td>66.3</td>
<td>30.1</td>
<td></td>
<td>29.9</td>
<td></td>
<td>66.3</td>
<td></td>
<td>30.1</td>
<td></td>
<td>29.9</td>
<td></td>
<td>81.1</td>
<td></td>
<td>80.9</td>
<td></td>
</tr>
<tr>
<td>Avg no. of trees</td>
<td>100</td>
<td>28</td>
<td>19</td>
<td></td>
<td>100</td>
<td>81.1</td>
<td>80.9</td>
<td></td>
<td>100</td>
<td></td>
<td>81.1</td>
<td></td>
<td>80.9</td>
<td></td>
<td>100</td>
<td></td>
<td>81.1</td>
<td></td>
<td>80.9</td>
<td></td>
</tr>
</tbody>
</table>
Table 3. Results for boosted decision trees and the logistic loss function.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>breastcancer</td>
<td>0.0351</td>
<td>0.0264</td>
<td>0.0264</td>
<td>0.00360</td>
<td>0.00400</td>
<td>0.00250</td>
</tr>
<tr>
<td></td>
<td>(0.0101)</td>
<td>(0.0120)</td>
<td>(0.00876)</td>
<td>(0.00100)</td>
<td>(0.00141)</td>
<td>(0.000866)</td>
</tr>
<tr>
<td>Avg tree size</td>
<td>15</td>
<td>59.9</td>
<td>14.0</td>
<td>15.0</td>
<td>7</td>
<td>22.1</td>
</tr>
<tr>
<td>Avg no. of trees</td>
<td>65.3</td>
<td>16.0</td>
<td>23.8</td>
<td>75.3</td>
<td>53.8</td>
<td>25.8</td>
</tr>
<tr>
<td>ionosphere</td>
<td>0.074</td>
<td>0.060</td>
<td>0.043</td>
<td>0.00360</td>
<td>0.00400</td>
<td>0.0170</td>
</tr>
<tr>
<td></td>
<td>(0.0236)</td>
<td>(0.0219)</td>
<td>(0.0188)</td>
<td>(0.00564)</td>
<td>(0.00245)</td>
<td>(0.00361)</td>
</tr>
<tr>
<td>Avg tree size</td>
<td>7</td>
<td>30.0</td>
<td>18.4</td>
<td>31.0</td>
<td>31.0</td>
<td>63.2</td>
</tr>
<tr>
<td>Avg no. of trees</td>
<td>100</td>
<td>25.3</td>
<td>29.5</td>
<td>100</td>
<td>54.0</td>
<td>37.0</td>
</tr>
<tr>
<td>german</td>
<td>0.233</td>
<td>0.232</td>
<td>0.225</td>
<td>0.00422</td>
<td>0.00417</td>
<td>0.00399</td>
</tr>
<tr>
<td></td>
<td>(0.0114)</td>
<td>(0.0123)</td>
<td>(0.0103)</td>
<td>(0.00191)</td>
<td>(0.00188)</td>
<td>(0.00211)</td>
</tr>
<tr>
<td>Avg tree size</td>
<td>7</td>
<td>7</td>
<td>14.4</td>
<td>15</td>
<td>15</td>
<td>25.9</td>
</tr>
<tr>
<td>Avg no. of trees</td>
<td>100</td>
<td>66.8</td>
<td>67.8</td>
<td>100</td>
<td>55.6</td>
<td>27.6</td>
</tr>
<tr>
<td>diabetes</td>
<td>0.250</td>
<td>0.246</td>
<td>0.246</td>
<td>0.00422</td>
<td>0.00417</td>
<td>0.0201</td>
</tr>
<tr>
<td></td>
<td>(0.0374)</td>
<td>(0.0356)</td>
<td>(0.0356)</td>
<td>(0.00412)</td>
<td>(0.00433)</td>
<td>(0.00411)</td>
</tr>
<tr>
<td>Avg tree size</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>28.7</td>
<td>33.5</td>
<td>72.8</td>
</tr>
<tr>
<td>Avg no. of trees</td>
<td>100</td>
<td>45.5</td>
<td>45.5</td>
<td>100</td>
<td>61.7</td>
<td>41.9</td>
</tr>
</tbody>
</table>
Margin Distribution

Ion: AdaBoost–L1, fold = 6
Ion: AdaBoost, fold = 6
Ion: DeepBoost, fold = 6
Cumulative Distribution of Margins
Extension to Multi-Class

- Similar data-dependent learning guarantee proven for the multi-class setting.
  - bound depending on mixture weights and complexity of sub-families.

- Deep Boosting algorithm for multi-class:
  - similar extension taking into account the complexities of sub-families.
  - several variants depending on number of classes.
  - different possible loss functions for each variant.
Multi-Class Learning Guarantee

Theorem: Fix $\rho > 0$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds for all $f = \sum_{t=1}^{T} \alpha_t h_t \in \mathcal{F}$:

$$R(f) \leq \hat{R}_{S,\rho}(f) + \frac{4c^2}{\rho} \sum_{t=1}^{T} \alpha_t \mathcal{R}_m(\Pi_1(H_{k_t})) + O\left(\sqrt{\frac{\log p}{\rho^2 m}} \log \left[\frac{\rho^2 m}{\log p}\right]\right),$$

with $c$ number of classes.

and $\Pi_1(H_{k_t}) = \{x \mapsto h(x, y) : y \in \mathcal{Y}, h \in H_k\}$. 
Conclusion

Deep Boosting: ensemble learning with increasingly complex families.

- data-dependent theoretical analysis.
- algorithm grounded in theory.
- extension to multi-class.
- ranking and other losses.
- enhancement of many existing algorithms.
- compares favorably to AdaBoost and AdaBoost-L1 in preliminary experiments.