Boosting Approaches to Learning on a Feature Budget

Lev Reyzin
UIC

Feature-Efficient Prediction Examples

- Medical testing

  Want to predict what patients are sick with, but tests might be expensive or dangerous.

- Displaying internet results

  Want to give users the best results you can, but if you don't give results within 300 milliseconds, users will leave.

Model

- Goal is to do supervised learning, using a limited number of features in test-time.
  - Given a budget on total cost: on each example, the learner must look at no more features than allowed by the budget.
  - Each feature has an associated cost.
  - Budget only limited in test data, not training.

- Predictors that do this are feature-efficient.

Lots of work on this problem

- Sequential analysis: when to stop sequential clinical trials. [Wald '47] and [Chernoff '72]
- PAC learning with incomplete features. [Ben-David-Dichterman '93] and [Greiner et al. '02]
- Robust prediction with missing features. [Globerson-Roweis '06]
- Learning linear functions by few features [Cesa-Bianchi et al. '10]
- Incorporating feature costs in CART impurity [Xu et al. '12]
- MDPs for feature selection [He et al. '13]

Idea: Use Ensembles / Boosting [R '11]

- An ensemble is usually a weighted vote of many simple rules.
- The simple rules are usually feature-efficient.
- If you choose the ensemble class properly and take a vote of only a few of the rules.

A Reminder of Boosting/ Ensembles

- Boosting combines many "moderately inaccurate" weak learners into an ensemble predictor.
- Generates a new weak learning on each round.
- Outputs a weighted vote of weak learners as classifier.

A Reminder of Boosting/ Ensembles

- Boosting combines many "moderately inaccurate" weak learners into an ensemble predictor.
- Generates a new weak learning on each round.
- Outputs a weighted vote of weak learners as classifier.

First Idea

- Sample from the distribution over weak learners.

AdaBoostRS [R '11]

Training: train AdaBoost (or any ensemble).

Prediction:

1. Sample the weak learners depending on their voting weights and feature costs.
2. Take a importance-weighted vote of the sampled weak learners.

Intuition:

If ensemble has strong preference, sampling will converge fast. If ensemble is split, who cares?

(resembles margin bound [Schapire et al. '98])

Training

- If all weak learners have same cost, just sample proportional to voting weights and take unweighted vote.
  - (can do this even with nonuniform costs)
- If each feature/weak learner has different cost, sample according to:

$$p(i) = \frac{\alpha_i}{c(h_i)\sum_{i=1}^{T}(\alpha_t / c(h_t))}$$

then de-bias in the voting

note: α are voting weights, c are costs

Margin Bounds

- AdaBoost:

$$\boldsymbol{P}_D[yf(x) \leq 0] \leq \boldsymbol{P}_S[yf(x) \leq \theta] + \tilde{O}\left(\sqrt{\frac{d}{m\theta^2}}\right)$$

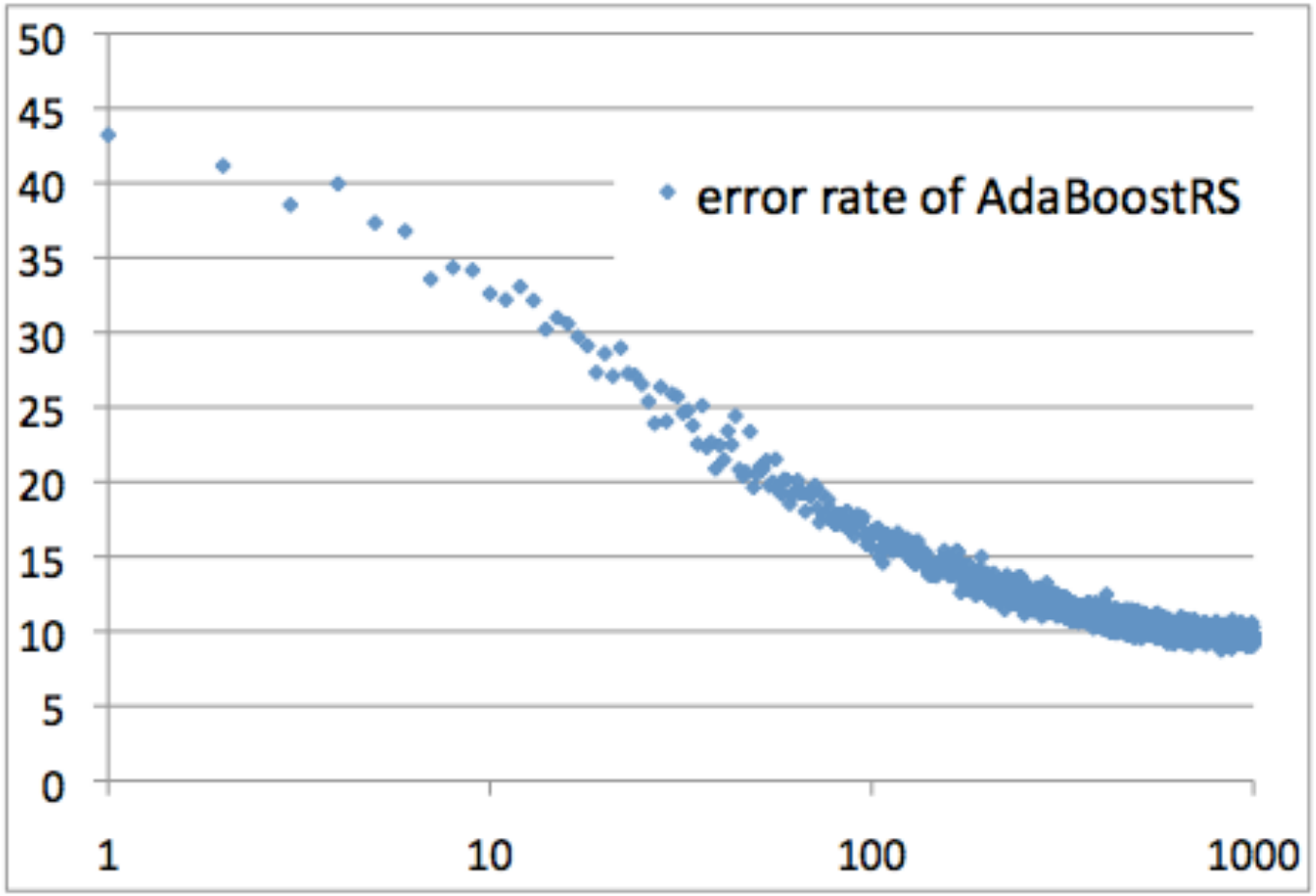
- AdaBoostRS:

$$\boldsymbol{P}_D[yf(x) \leq 0] \quad \leq \quad \boldsymbol{P}_S[yf(x) \leq \theta] + \tilde{O}\left(\sqrt{\frac{d}{m\theta^2}}\right) + e^{-N\theta^2/2}$$

(also birthday paradox helps)

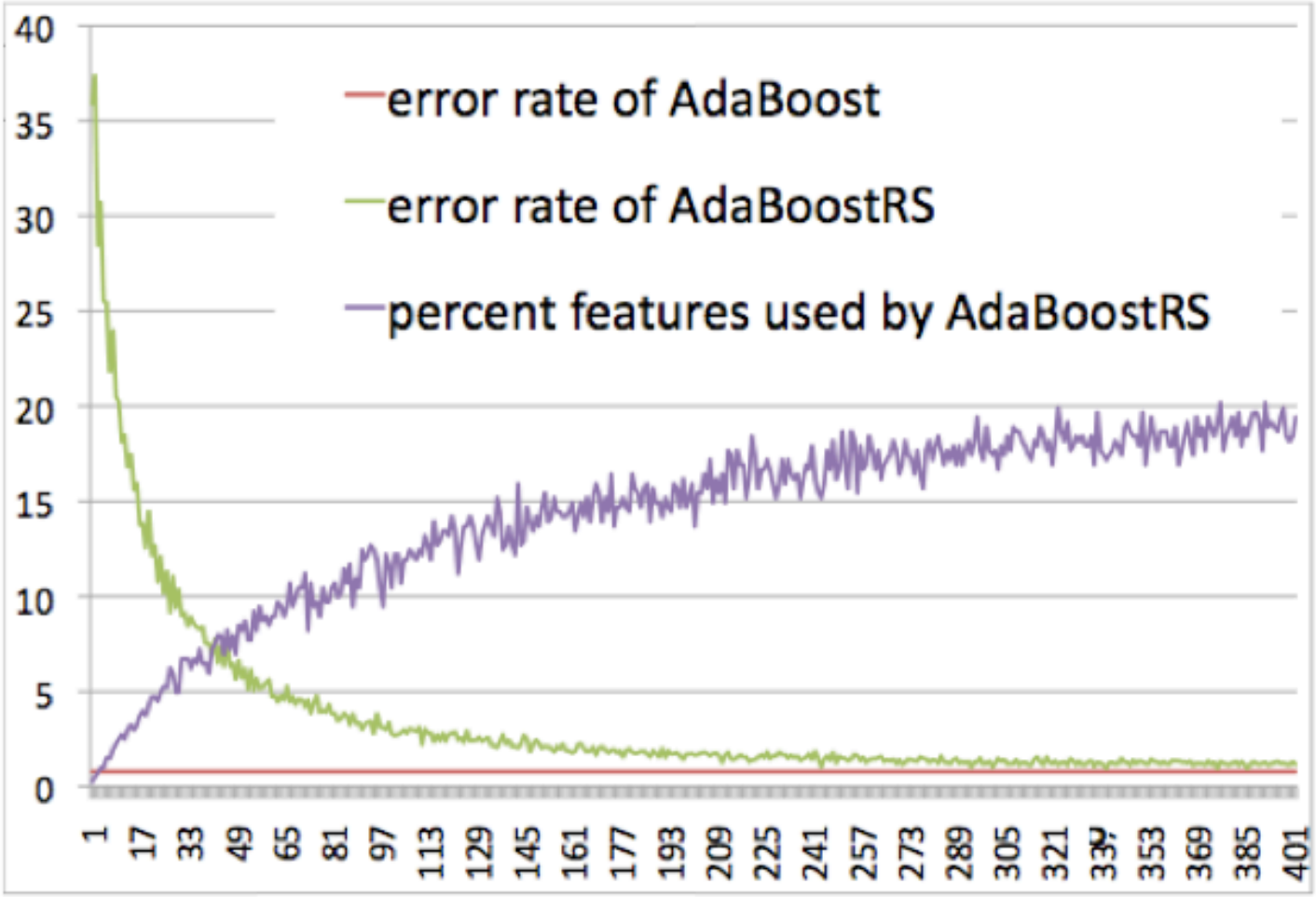
d is VC dimension, m is number of training examples, N is number of (weak learner) samples AdaBoostRS takes

Experiments with AdaBoostRS



A graph of the error rate of **AdaBoostRS** on the splice dataset, as a function of the number of samples. The horizontal axis is on a log scale.

Experiments with AdaBoostRS



On ocr17 dataset. x-axis is number of samples taken.

Taking Costs into Account (explanation)

Number of samples taken (τ), averaged over 50 trials, of $\texttt{AdaBoostRS}_{\text{AC}}$ and AdaBoostRS using budgets of 11 and 21 when features have random costs drawn i.i.d. from $[0, 1]$. The underlying algorithm, AdaBoost, is run for 500 rounds.

	$\texttt{AdaBoostRS}_{\text{AC}}$	AdaBoostRS
census ($B = 11$)	26.2	20.7
census ($B = 21$)	45.7	41.3
splice ($B = 11$)	33.8	20.6
splice ($B = 21$)	56.2	40.0
ocr17 ($B = 11$)	29.4	20.6
ocr17 ($B = 21$)	49.3	40.5
ocr49 ($B = 11$)	33.6	21.1
ocr49 ($B = 21$)	55.7	40.3

Taking Costs into Account (error rates)

Error rates (in percent), averaged over 50 trials, of **AdaBoostRS$_{AC}$** and **AdaBoostRS** using budgets of 10 and 20 when features have random costs drawn i.i.d. from $[0, 1]$. The underlying algorithm, **AdaBoost**, is run for 500 rounds.

	AdaBoostRS$_{AC}$	**AdaBoostRS**
census $(B = 11)$	32.2	32.8
census $(B = 21)$	25.5	26.4
splice $(B = 11)$	25.7	27.0
splice $(B = 21)$	19.2	20.4
ocr17 $(B = 11)$	9.2	10.5
ocr17 $(B = 21)$	3.5	4.3
ocr49 $(B = 11)$	27.4	28.3
ocr49 $(B = 21)$	20.2	21.4

Room for Improvement?

Pro: works on any ensemble

Con: budget not considered in training.

So, can we improve by moving the optimization into training?

Turns out: yes, by a lot! [Huang-Powers-R '14]

- Naïve idea: train AdaBoost until budget runs out
- Improvement: choose weak learners more wisely

AdaBoost (S) where: $S \subset X \times \{-1, +1\}$

1: given: $(x_1, y_1), \ldots, (x_m, y_m) \in S$
2: initialize $D_1(i) = \frac{1}{m}$
3: **for** $t = 1, \ldots, T$ **do**
4: train base learner using distribution D_t.
5: get $h_t \in \mathcal{H} : X \to \{-1, +1\}$.

6: choose $\alpha_t = \frac{1}{2} \ln \frac{1+\gamma_t}{1-\gamma_t}$, where $\gamma_t = \sum_i D_t(i) y_i h_t(x_i)$.
7: update $D_{t+1}(i) = D_t(i) \exp(\alpha_t y_i h_t(x_i))/Z_t$,
8: **end for**
9: output the final classifier $H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$

AdaBoostBT(S,B,C) where: $S \subset X \times \{-1,+1\}$, $B > 0$, $C : [n] \to \mathbb{R}^+$

given: $(x_1, y_1), \ldots, (x_m, y_m) \in S$
initialize $D_1(i) = \frac{1}{m}$, $B_1 = B$
for $t = 1, \ldots, T$ **do**
  train base learner using distribution D_t.
  get $h_t \in \mathcal{H} : X \to \{-1,+1\}$.
  if the total cost of the unpaid features of h_t exceeds B_t **then**
    set $T = t - 1$ and **end for**
  else set B_{t+1} as B_t minus the total cost of the unpaid features of h_t, marking them as paid
  choose $\alpha_t = \frac{1}{2}\ln\frac{1+\gamma_t}{1-\gamma_t}$, where $\gamma_t = \sum_i D_t(i) y_i h_t(x_i)$.
  update $D_{t+1}(i) = D_t(i)\exp(\alpha_t y_i h_t(x_i))/Z_t$,
end for
output the final classifier $H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$

How to choose weak learner h_t?

Training error of AdaBoost is bounded by
[Freund & Schapire '97]

$$\hat{\Pr}[H(x) \neq y] \leq \prod_{t=1}^{T} \sqrt{1-\gamma_t^2}$$

With budgets, we need to consider two effects:

- high edges make individual terms smaller
- low costs allow for more terms in the product

Two Optimizations
[Huang-Powers-R '14]

First idea: assume all future rounds will behave like current. Leads to optimization

1) $$h_t = \text{argmin}_{h \in \mathcal{H}} \left((1 - \gamma_t(h)^2)^{\frac{1}{c(h)}} \right)$$

Second idea: smoothed version of first.

2) $$h_t = \text{argmin}_{h \in \mathcal{H}} \left((1 - \gamma_t(h)^2)^{\frac{1}{(B - B_t) + c(h)}} \right)$$

A Competing Approach: SpeedBoost [Grubb-Bagnell '12]

SpeedBoost:

Define objective R (eg exponential loss)

while not over budget {

let h, α = argmax = $R[f_{i-1}] + R[f_{i-1} + \alpha h] / c(h)$

}

A Competing Approach: SpeedBoost [Grubb-Bagnell '12]

SpeedBoost:

Define objective R (eg exponential loss)

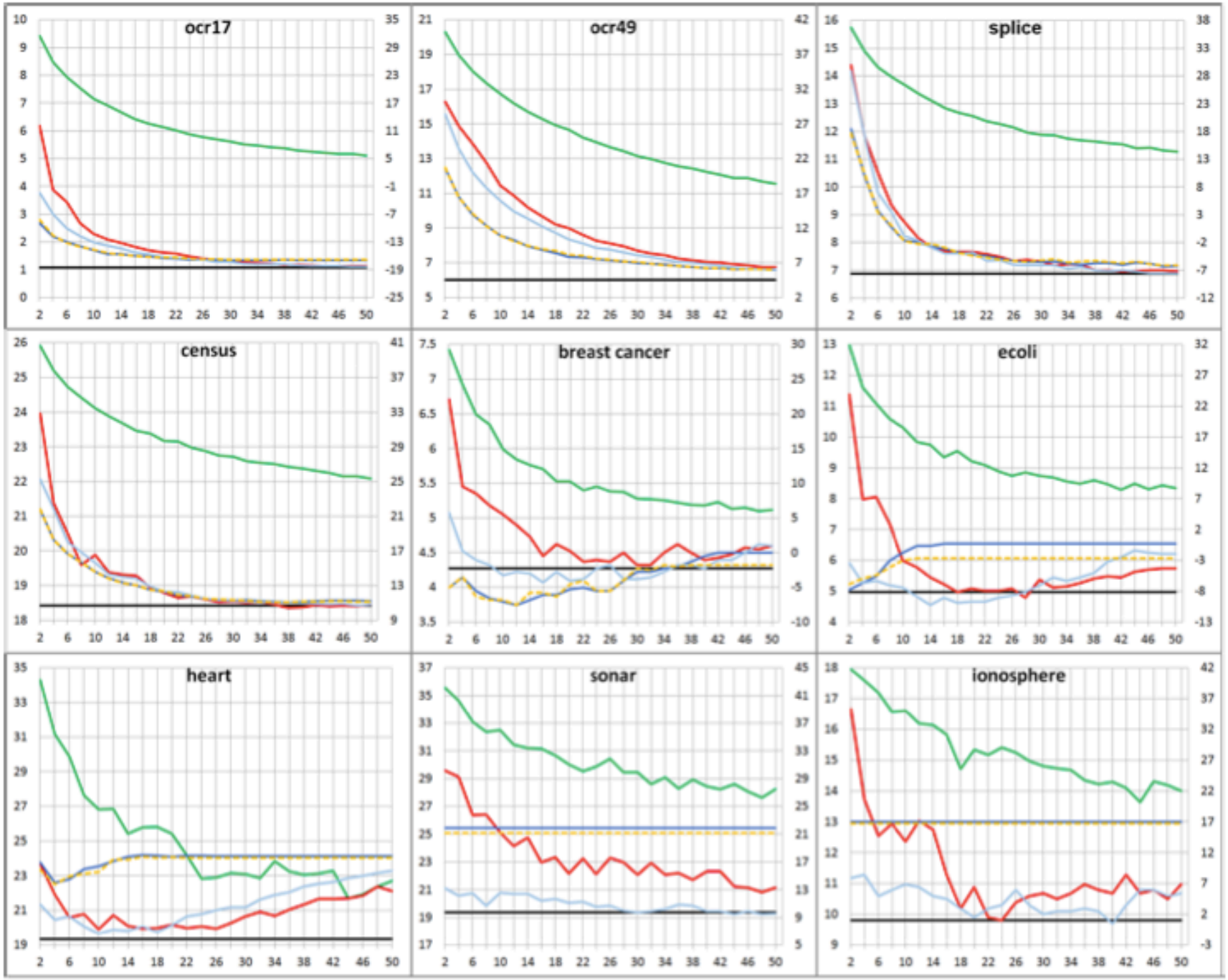
while not over budget {

let h, α = argmax = $R[f_{i-1}] + R[f_{i-1} + \alpha h] / c(h)$

}

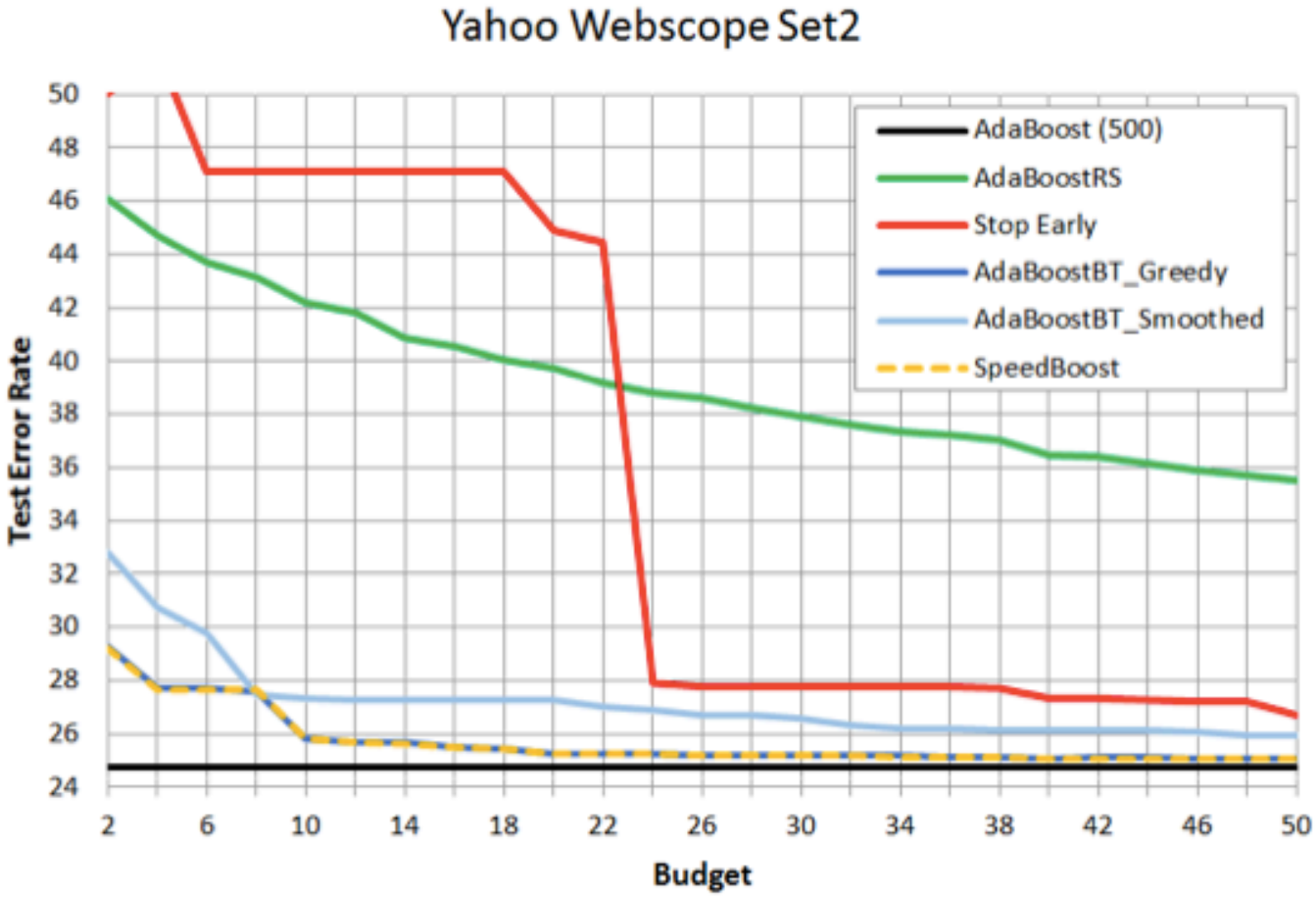
Tractable?

Note: exponential loss and uniform cost give AdaBoost.



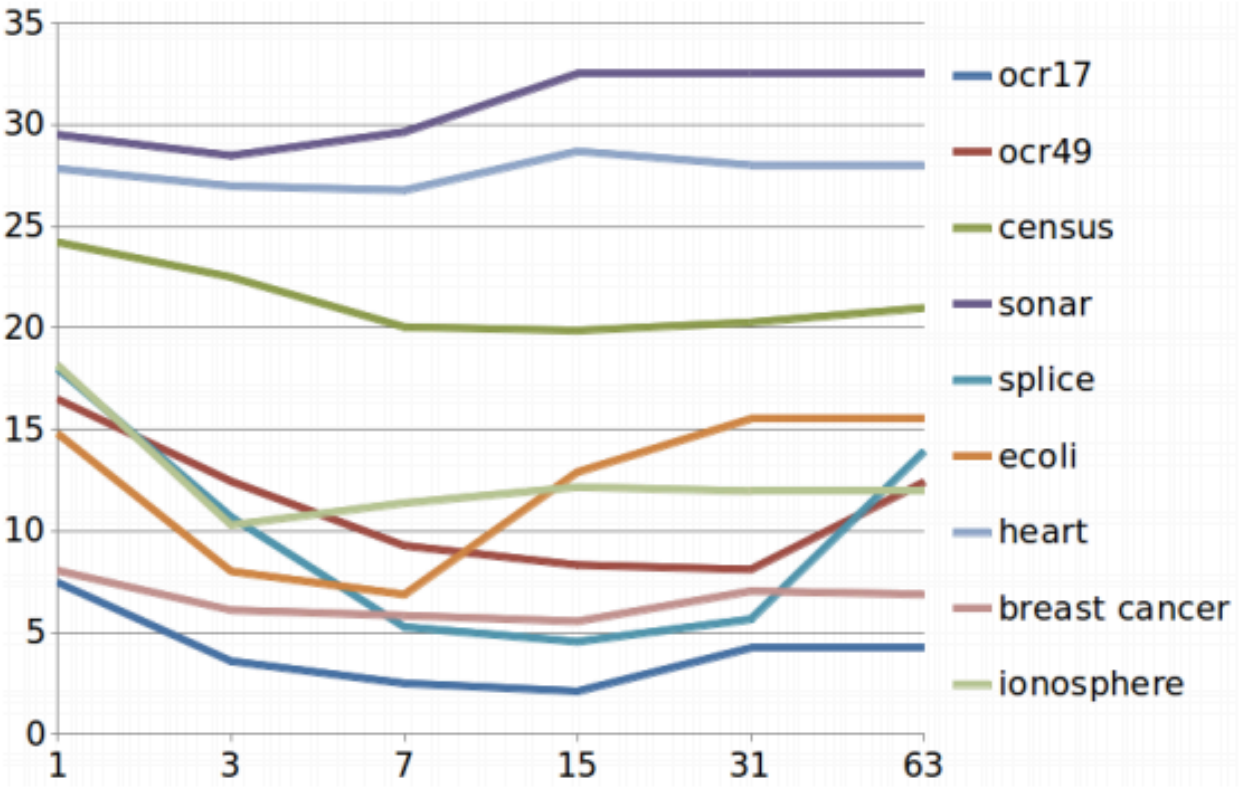
Experiments with costs ~ U(0,2)

On Yahoo! Data (with real costs)



Experimental results comparing our approaches to `AdaBoostRS` and `SpeedBoost` on the Yahoo! Webscope data set 2. Test error is calculated at budget increments of 2.

Finally… Decision Trees



Error Rates of decision trees. The horizontal is these number of nodes (log scale in number of nodes, linear in expected tree depth). The vertical is percent error.

Discussion

Ensembles seem well suited for predicting on a feature budget.

Find better ensemble algorithms?

Better feature-efficient weak learners? (currently working on this [R '14])

How do these compare with other approaches?