Sampling Strategies for Feature-Efficient and Active Learning

Lev Reyzin
UIC, Math Department

Feature-Efficient Prediction

with

Yi Huang and Brian Powers

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

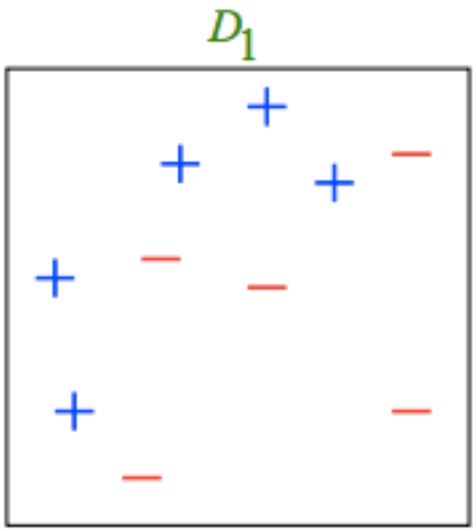
- Sequential analysis: when to stop sequential clinical trials.
  [Wald '47] and [Chernoff '72]
- PAC learning with incomplete features.
  [Ben-David-Dichterman '93] and [Greiner et al. '02]
- Robust prediction with missing features.
  [Globerson-Roweis '06]
- Learning linear functions by few features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

A Sampling Idea [R '11]

- An ensemble is usually a weighted vote of many simple rules.
- The simple rules are usually feature-efficient.
- Take a vote of only a few of the rules.

AdaBoost in Pictures (Slides from Schapire)

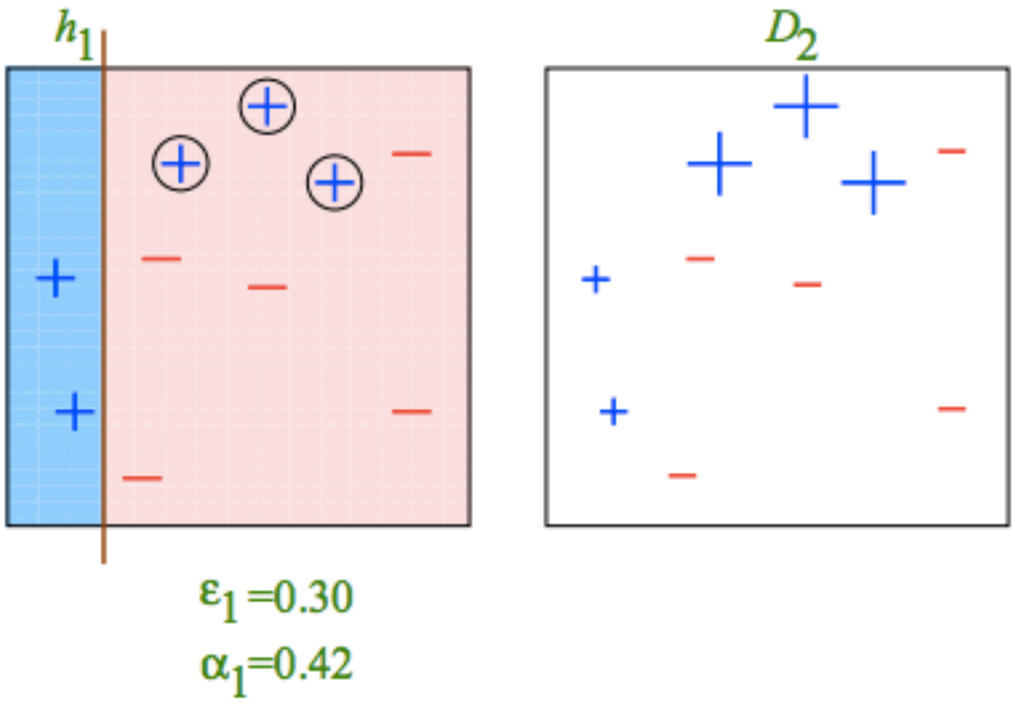
Toy Example



weak classifiers = vertical or horizontal half-planes

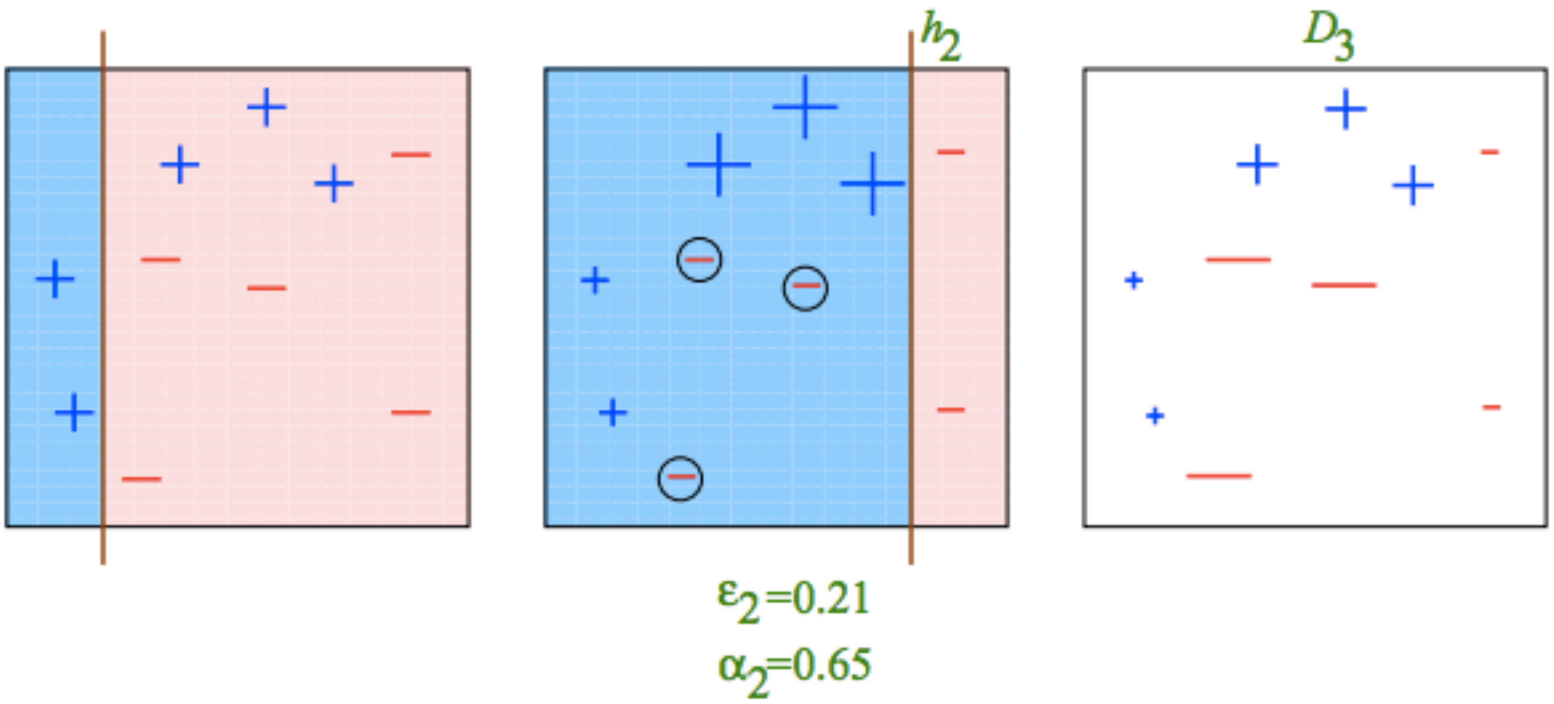
AdaBoost in Pictures (Slides from Schapire)

Round 1



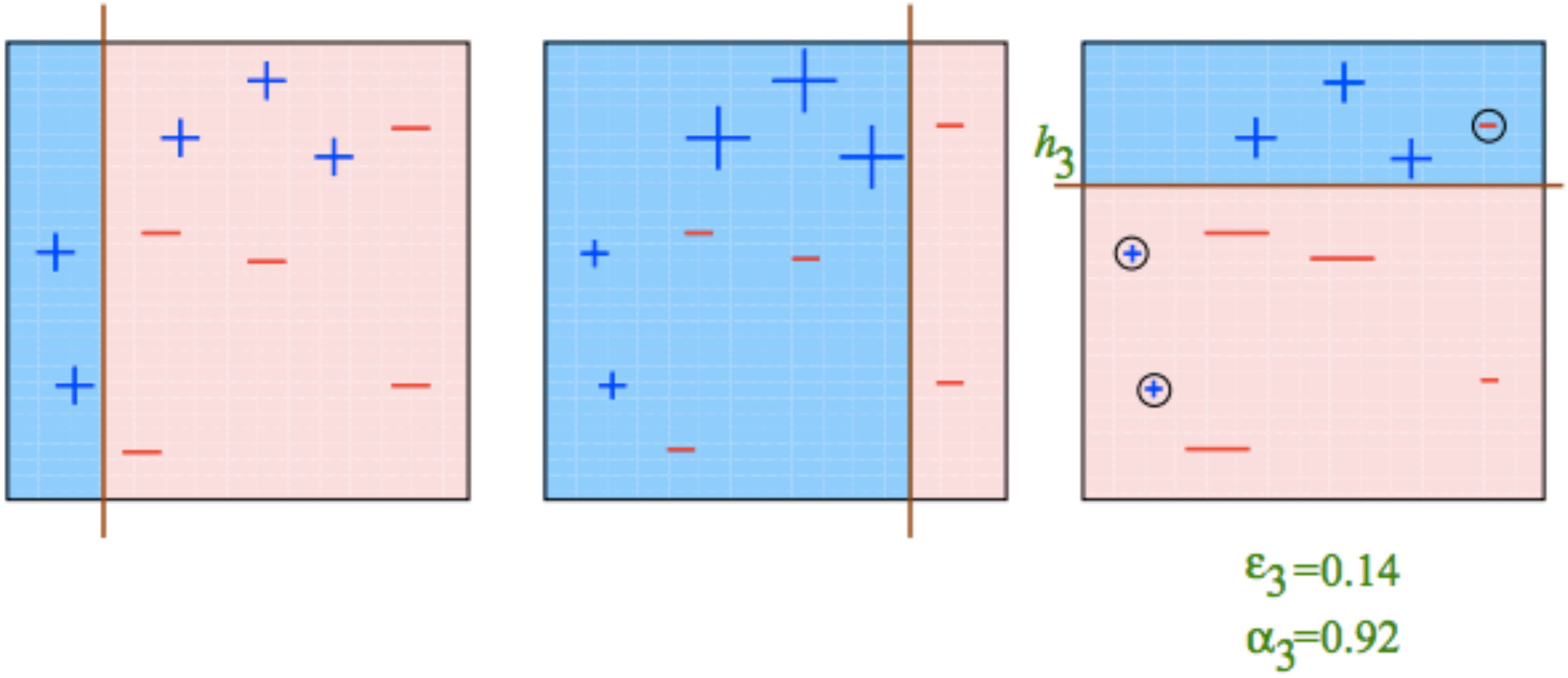
AdaBoost in Pictures (Slides from Schapire)

Round 2



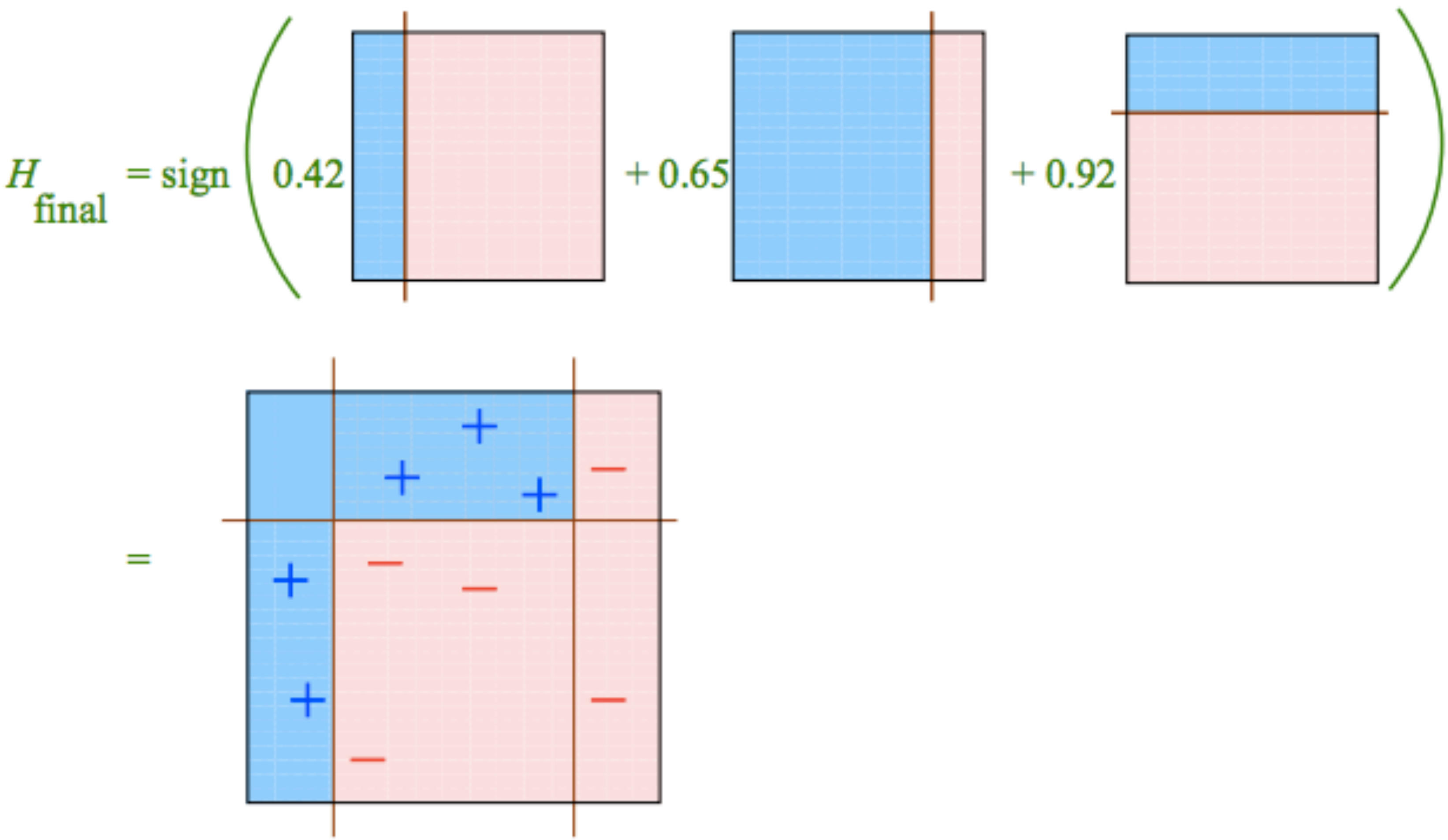
AdaBoost in Pictures (Slides from Schapire)

Round 3



AdaBoost in Pictures (Slides from Schapire)

Final Classifier



AdaBoostRS [R '11]

Training: train AdaBoost (or any ensemble).

Prediction:

1. Sample the weak learners depending on their voting weights and feature costs.

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

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

Bound:

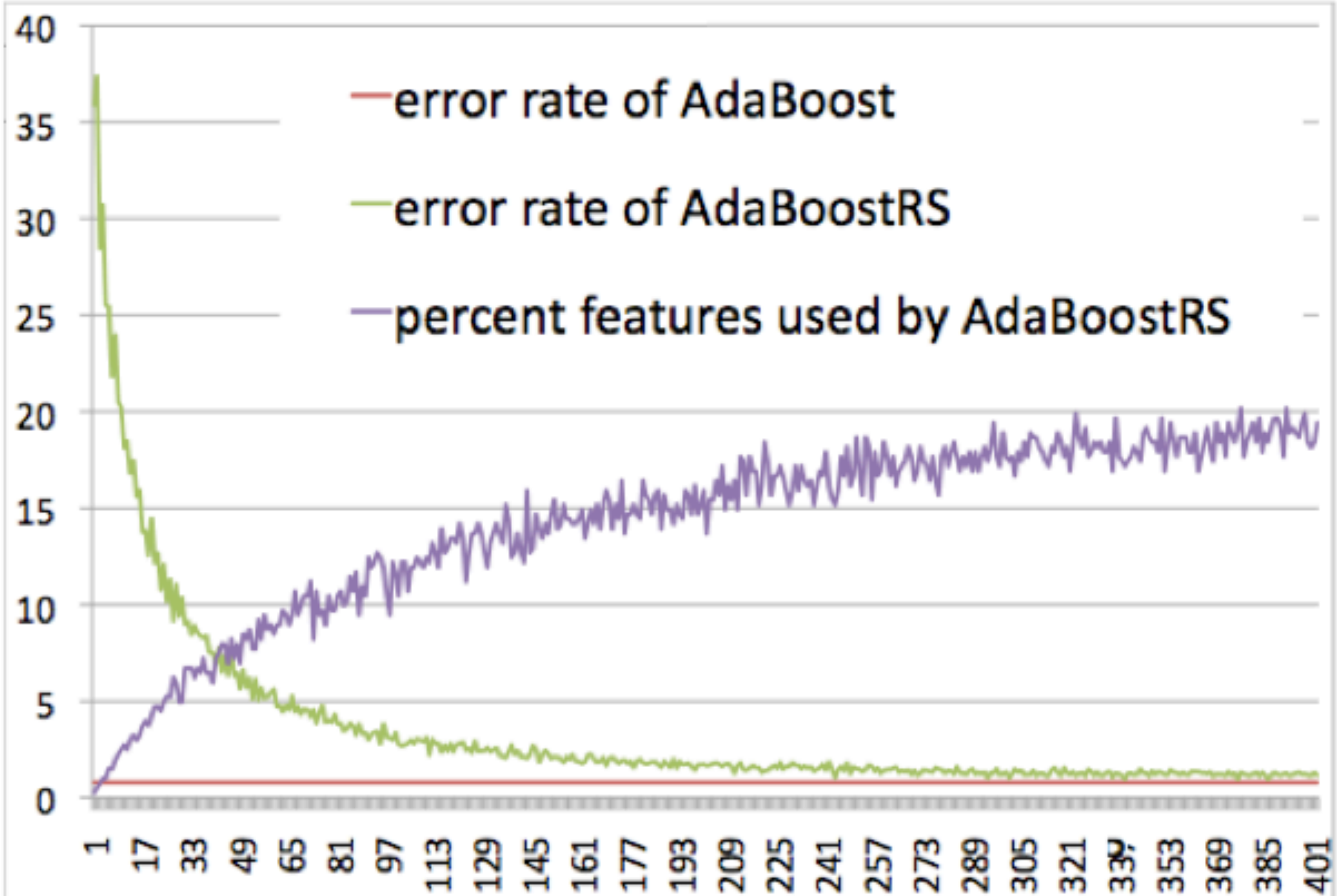
AdaBoost:

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

AdaBoostRS:

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

Experiments with AdaBoostRS



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

Room for Improvement

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: $S \subset X \times \{-1,+1\}$

given: $(x_1, y_1), \ldots, (x_m, y_m) \in S$
initialize $D_1(i) = \frac{1}{m}$
for $t = 1, \ldots, T$ **do**
    train base learner using distribution D_t.
    get $h_t \in \mathcal{H} : X \to \{-1,+1\}$.
    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 $H(x) = \mathrm{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$

`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]

<u>First idea</u>: assume all future rounds will behave like current. Leads to optimization

So $T = \frac{B}{c(h)}$.

Then the selection becomes

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

Two Optimizations

[Huang-Powers-**R** '14]

Second idea: cost of future rounds is average cost so far

The resulting selection rule is

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

Idea: Using average cost should produce a smoother optimization.

SpeedBoost
[Grubb-Bagnell '12]

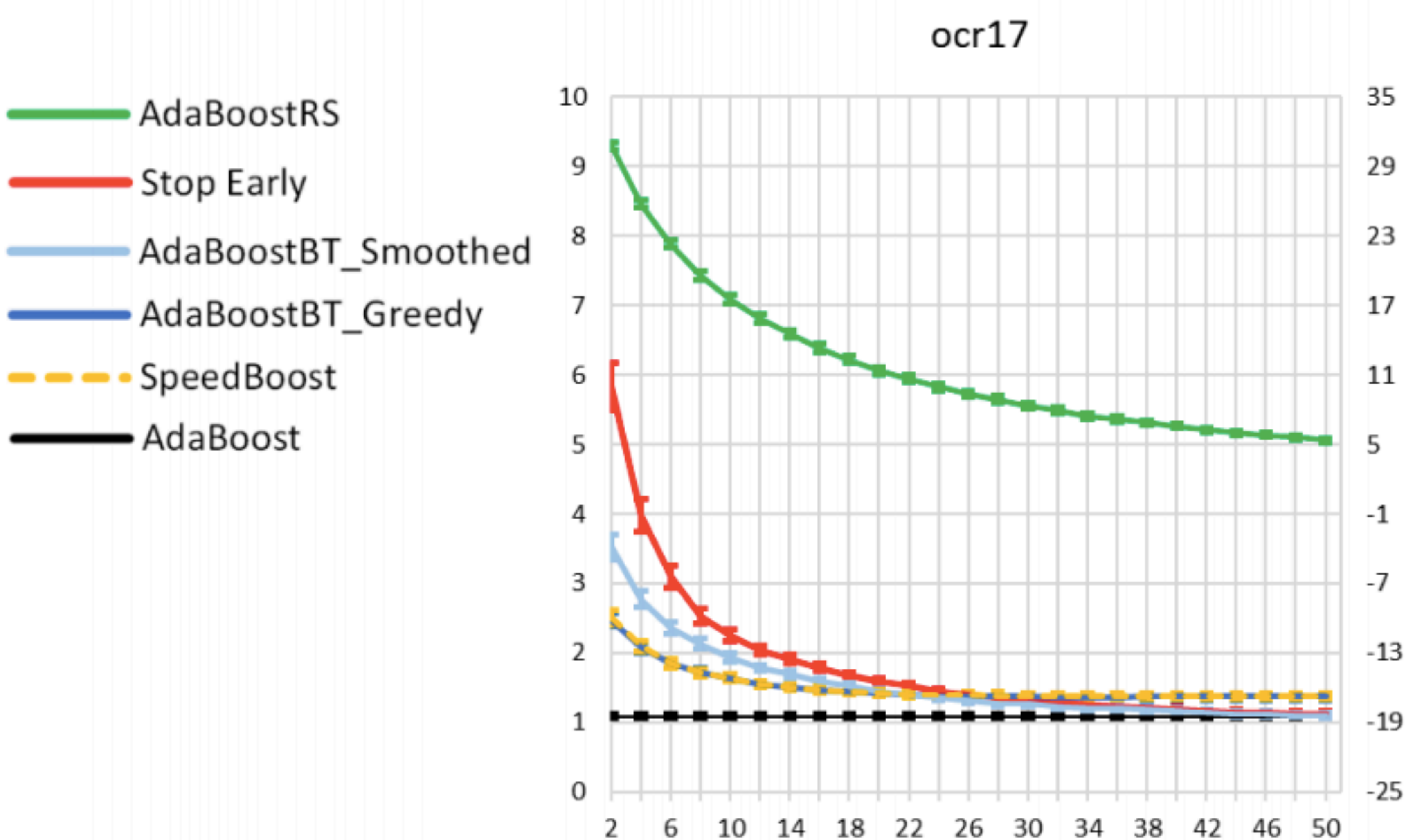
SpeedBoost [Grubb-Bagnell '12] produces a feature-efficient ensemble in another way.

An objective R is chosen (e.g. a loss function).

While the budget allows:
A Weak learner h and weight α are chosen to maximize

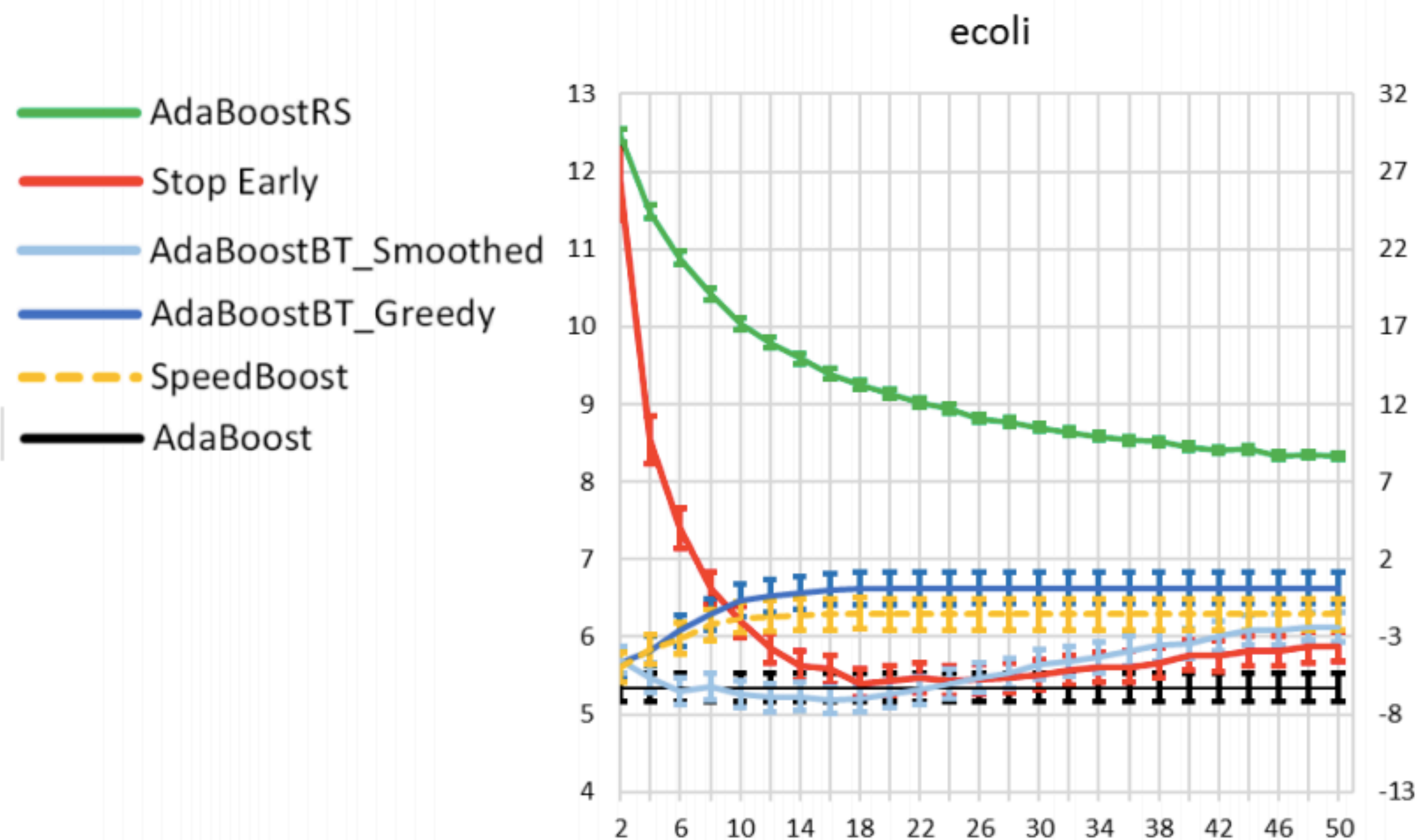
$$\frac{R(f_{i-1}) - R(f_{i-1} + \alpha h)}{c(h)}.$$

Experiments with costs ~ U(0,2)

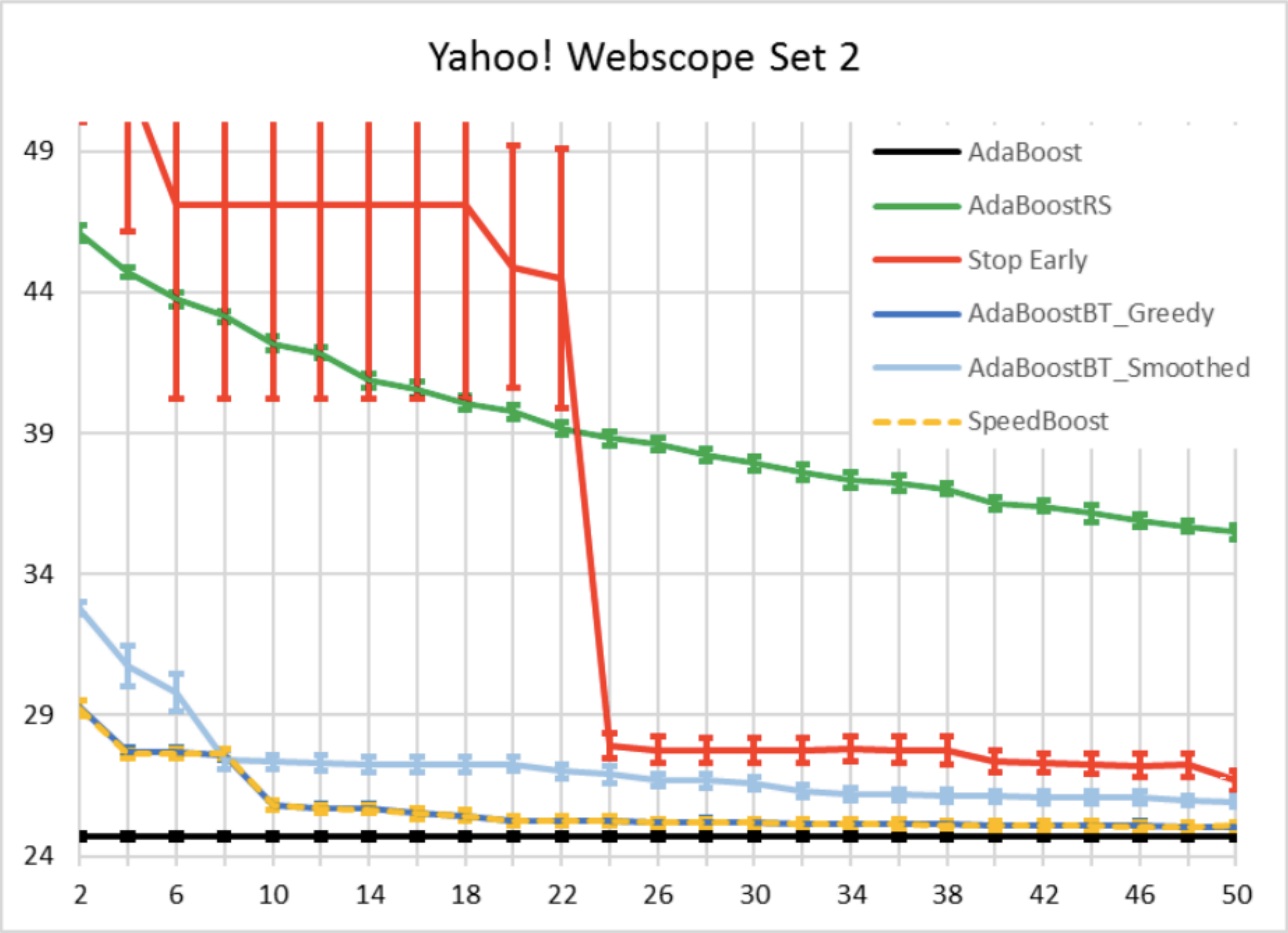


Budget on horizontal axis, test error rate on vertical (AdaBoostRS error on right). AdaBoost at T=400 as a benchmark.

Experiments with costs ~ U(0,2)



Budget on horizontal axis, test error rate on vertical (AdaBoostRS error on right). AdaBoost at T=400 as a benchmark.


Yahoo! Webscope Set 2
49
44
39
34
29
24
2
6
10
14
18
22
26
30
34
38
42
46
50
AdaBoost
AdaBoostRS
Stop Early
AdaBoostBT_Greedy
AdaBoostBT_Smoothed
SpeedBoost

- We gave a simple, generic way of choosing weak learners for budgeted learning.
  - Speedboost basically turns out to be a special case.

- Perhaps, better yet, would be to dynamically choose a tradeoff function as boosting continues.
  - We don't know how to do this yet.

- Need to compare to the wide variety of other techniques (MDPs [He et al. '13], decision trees with impurity [Xu et al. '12], etc.)

Active Learning

with

Anqi Liu and Brian Ziebart

Pool-Based Active Learning

- A pool based active learning algorithm sequentially chooses which labels to solicit from a pool of examples. [Lewis-Gale '94]
  - Usually constructs estimate of conditional label distribution $P(y|x)$ from labeled dataset.
  - Uses own estimate to select next datapoint label.

(I'll focus on logloss, but ideas are more general)

Uncertainty Sampling

- Many active learning strategies employ uncertainty sampling – selecting examples about which the algorithm is least certain.
- Other strategies assess how a label:
  - is expected to change model [Settles-Craven '08]
  - reduces an upper bound on the generalization error in expectation [Mackay '92]
  - represents the input patterns of remaining unlabeled data [Settles '12]

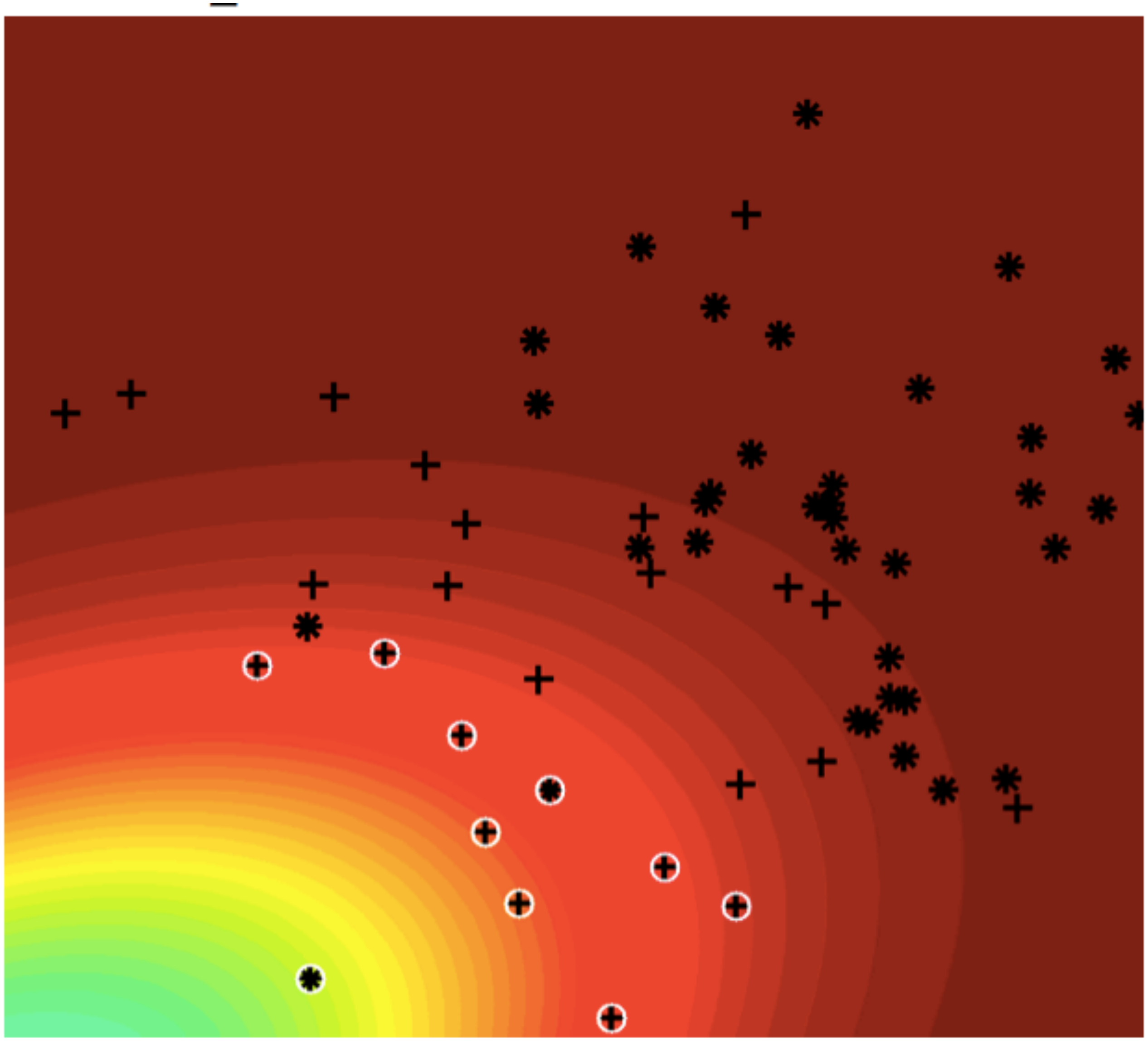
A Problem

Current active learning algorithms often perform poorly in practice [Attenberg-Provost '11].

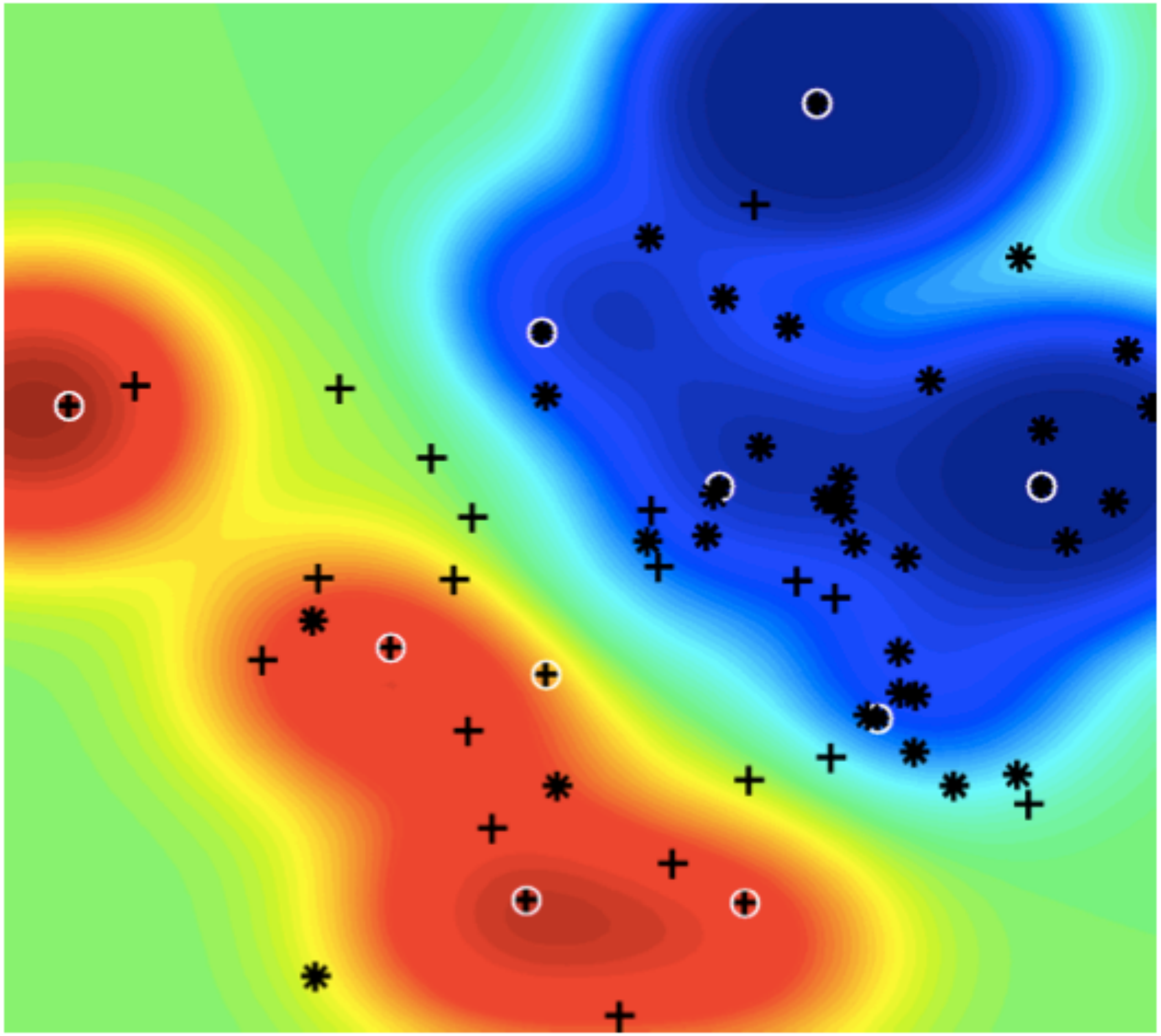
Why?

- In order to be take advantage of active learning, a biased label solicitation strategy should be used.
- Most current active learning strategies are overconfident, given this bias.

Typical Active Learner Behavior



Desired Behavior



Some Attempts to Fix This

- Seeding the active learner with a small random set [Dligach-Palmer '11].
- Restricting the active learner to a small set of examples [Schein-Ungar '07].
- Etc.

However, these modifications treat the symptoms of optimistic modeling and biased sampling and restrict the active learner, undermining its benefit.

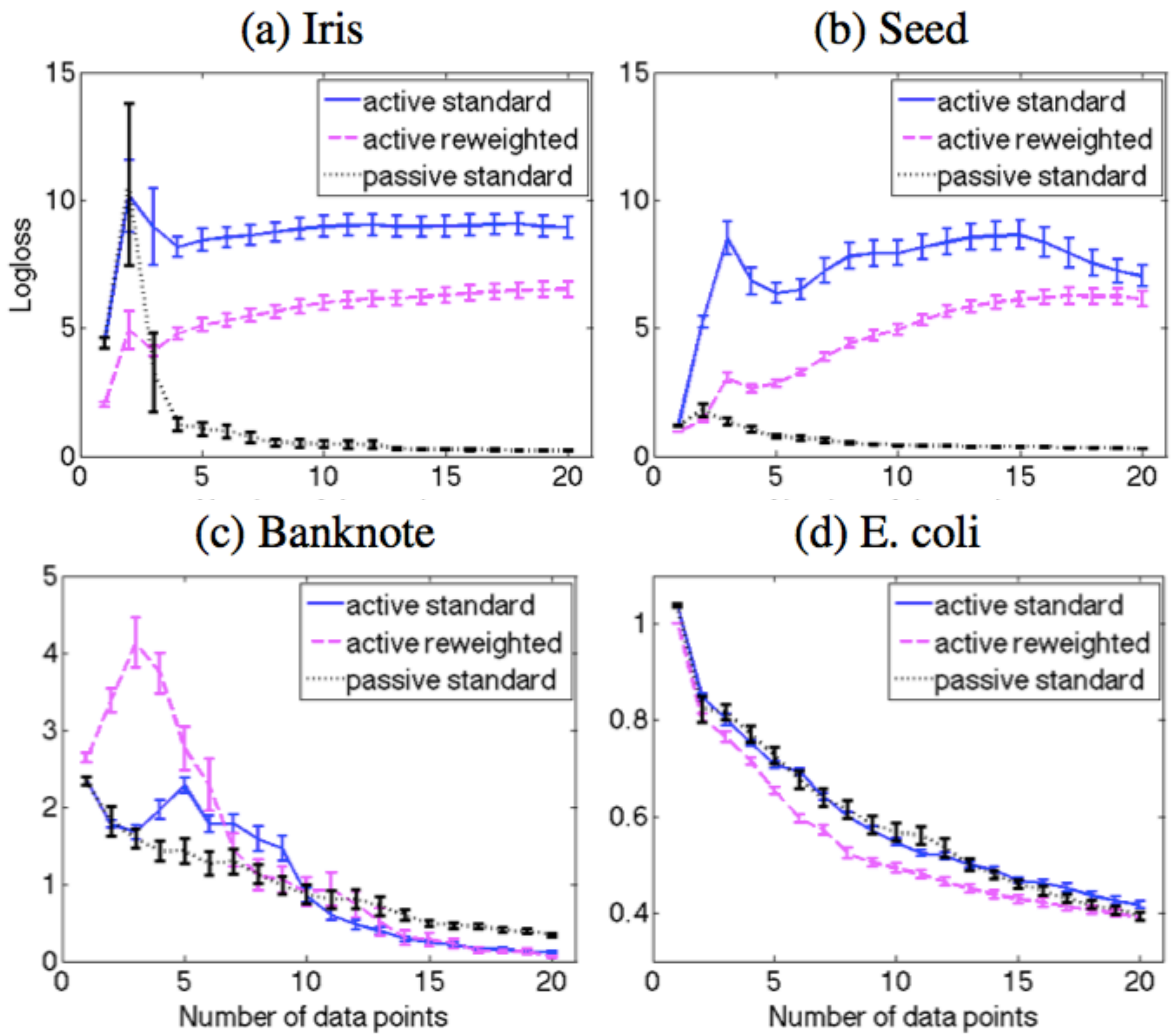
Biased Label Solicitation

Key idea: Active learning always leads to sample selection bias exists. Here, known as covariate shift -- P(Y | X) is shared in source and target distributions.

Tackling covariate shift is difficult. A common approach is importance re-weighting of source samples x according to $P_{trg}(x)/P_{src}(x)$ and minimizing a reweighted version of the loss [Shimodaria '00].

This converges slowly [Cortes-Mansour-Mohri '10] and the variance of estimates is too high to be useful.

Logistic Regression Models



Approach

- We use the recently developed RBA (robust bias-aware prediction) framework for tackling covariate shift [Liu-Ziebart '14].
- RBA solves a game against a constrained adversary that chooses an evaluation distribution:

$$\min_{\hat{P}(y|x)} \max_{\check{P}(y|x) \in \tilde{\Xi}} \mathbb{E}_{P_{\mathcal{D}}(x)\check{P}(y|x)} \overbrace{[-\log \hat{P}(Y|X)]}^{\text{logarithmic loss}}$$

The set $\tilde{\Xi}$ constrains the adversary

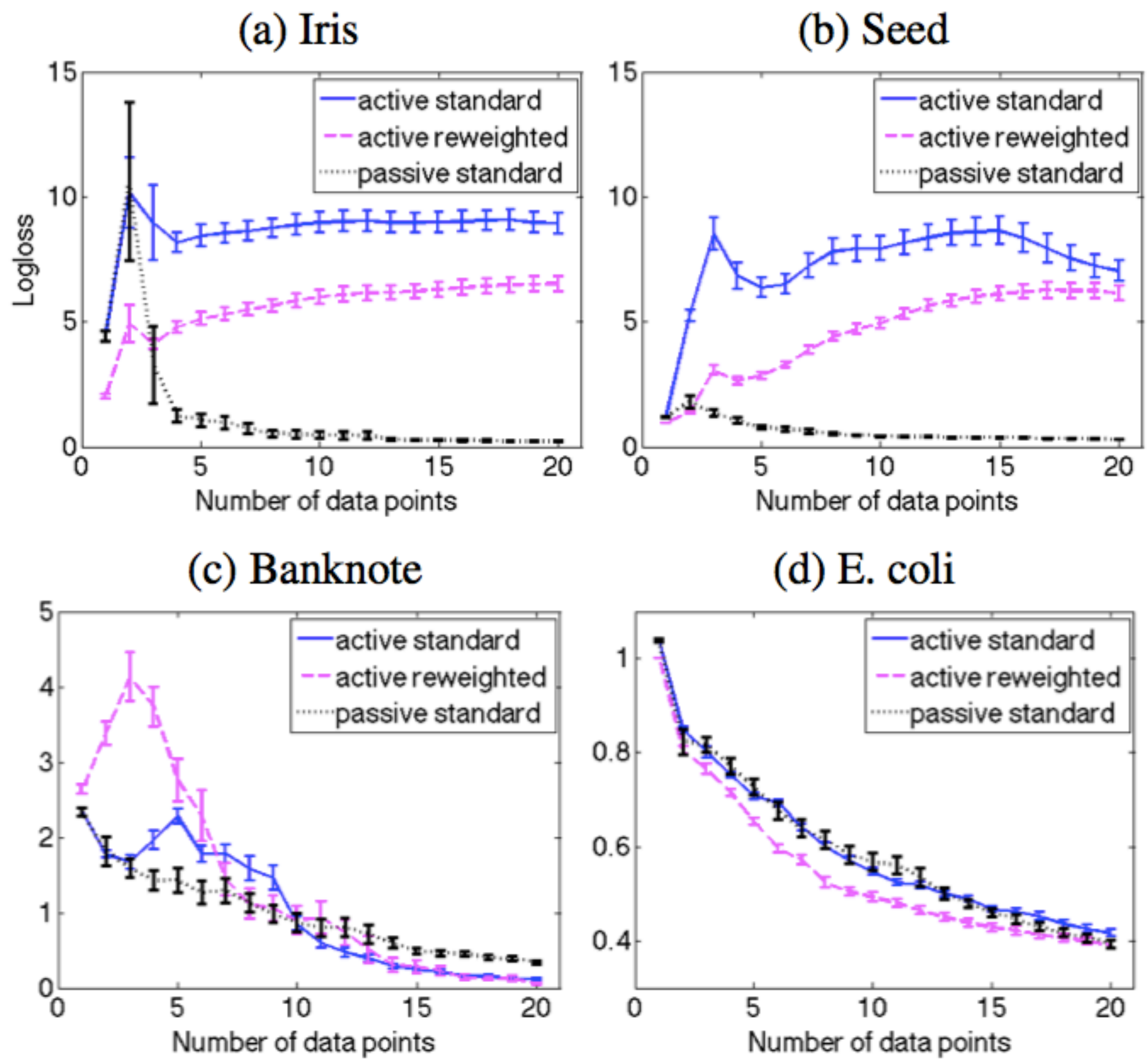
Robust Prediction Strategy

- The RBA predictor can be obtained by solving the dual of a conditional max entropy estimation problem. [Liu-Ziebart '14]
- Can be shown to upper bound the the generalization loss, under some assumptions. [Grunwald-Dawid '04]
- $P_{src}(x)$ needs to be estimated – we use kernel density estimation with Gaussian kernels for $P_{src}(x)$.
- The RBA predictor turns out to less certain where the labeled data underrepresents the full data distribution.

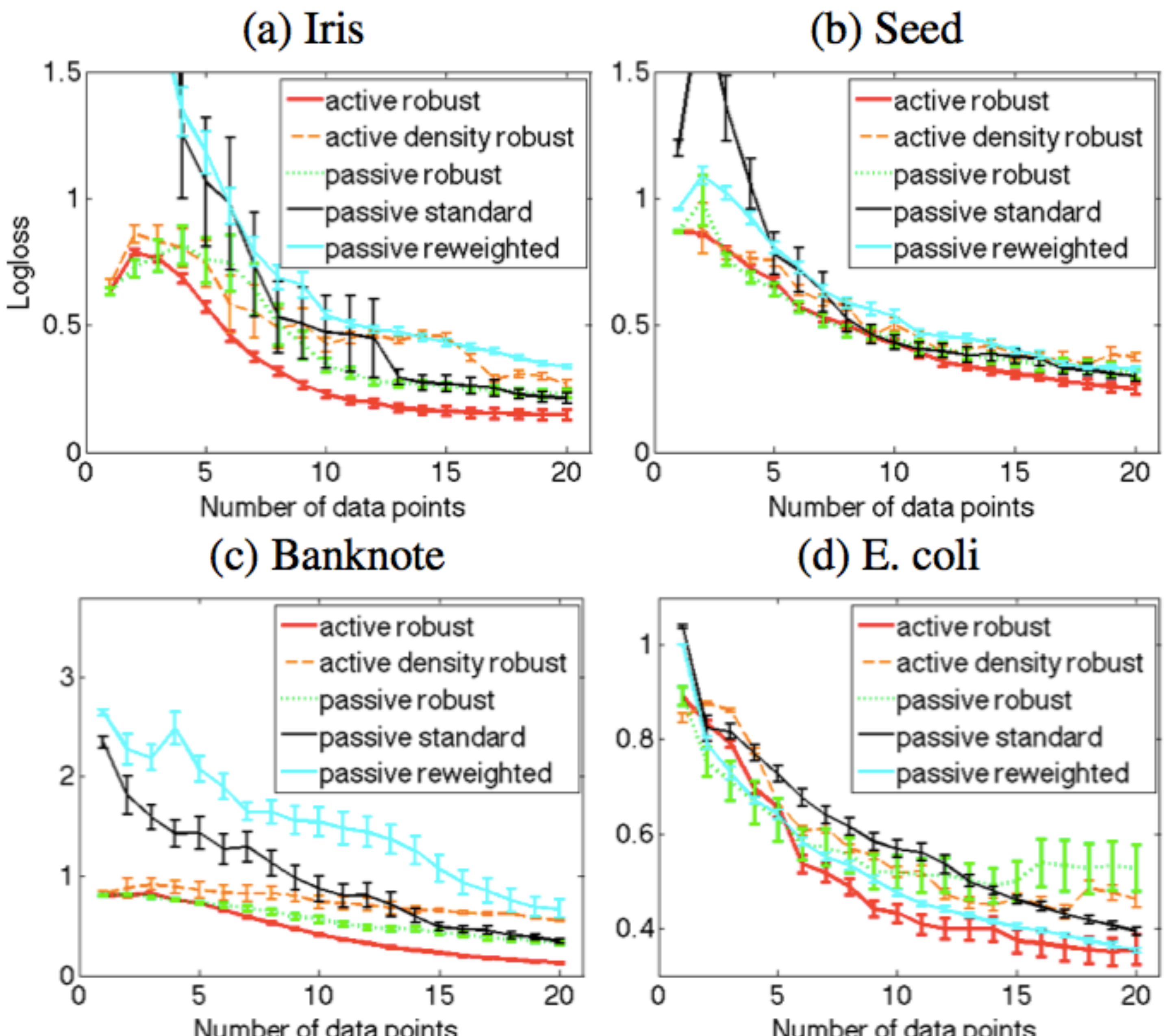
Sampling Strategies

- active robust – select point with largest value conditioned entropy
- active random – select point at random
- active density – select point with highest density ratio of $P_D(x)/P_L(x)$

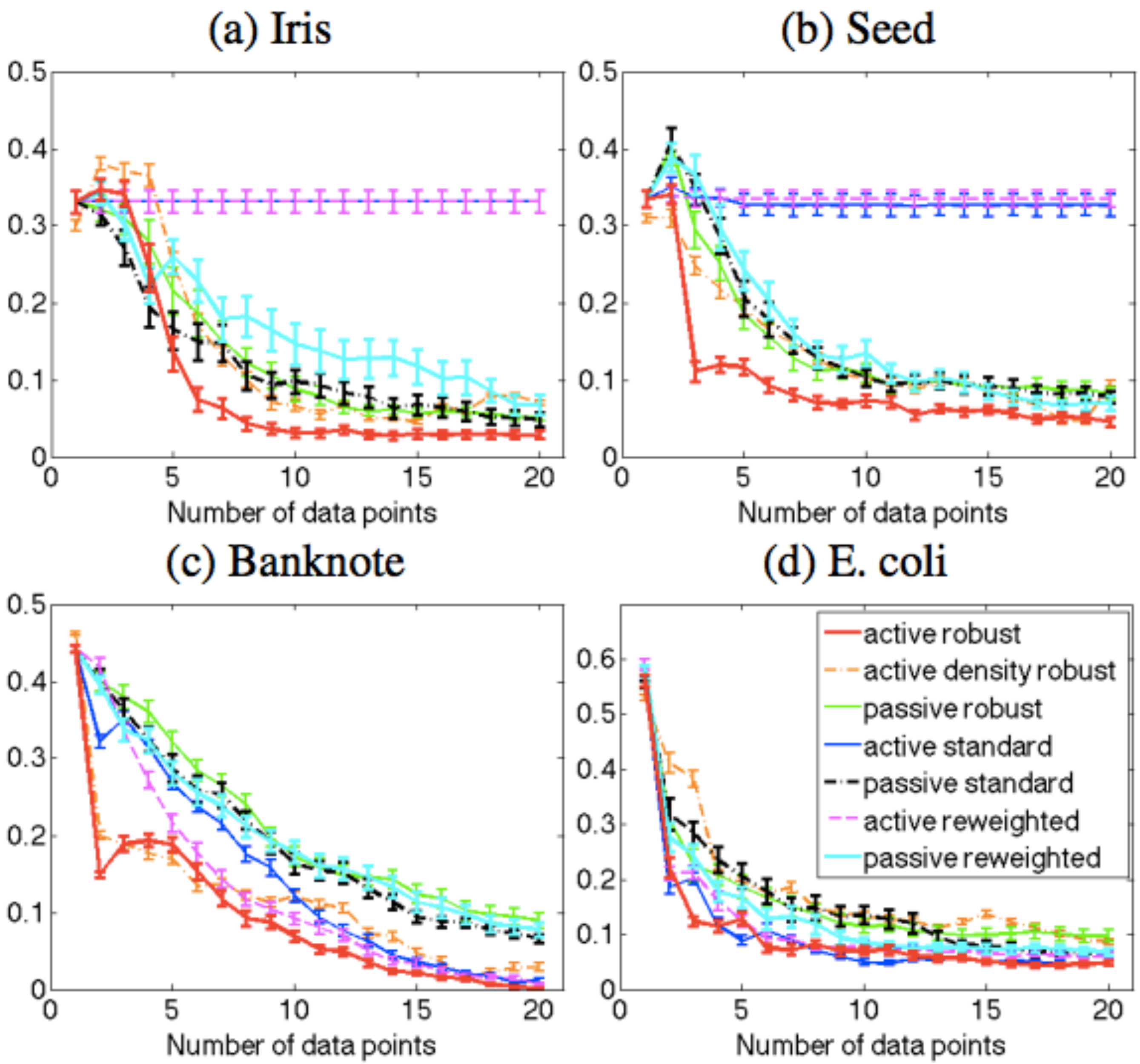
Standard Logistic Regression Models



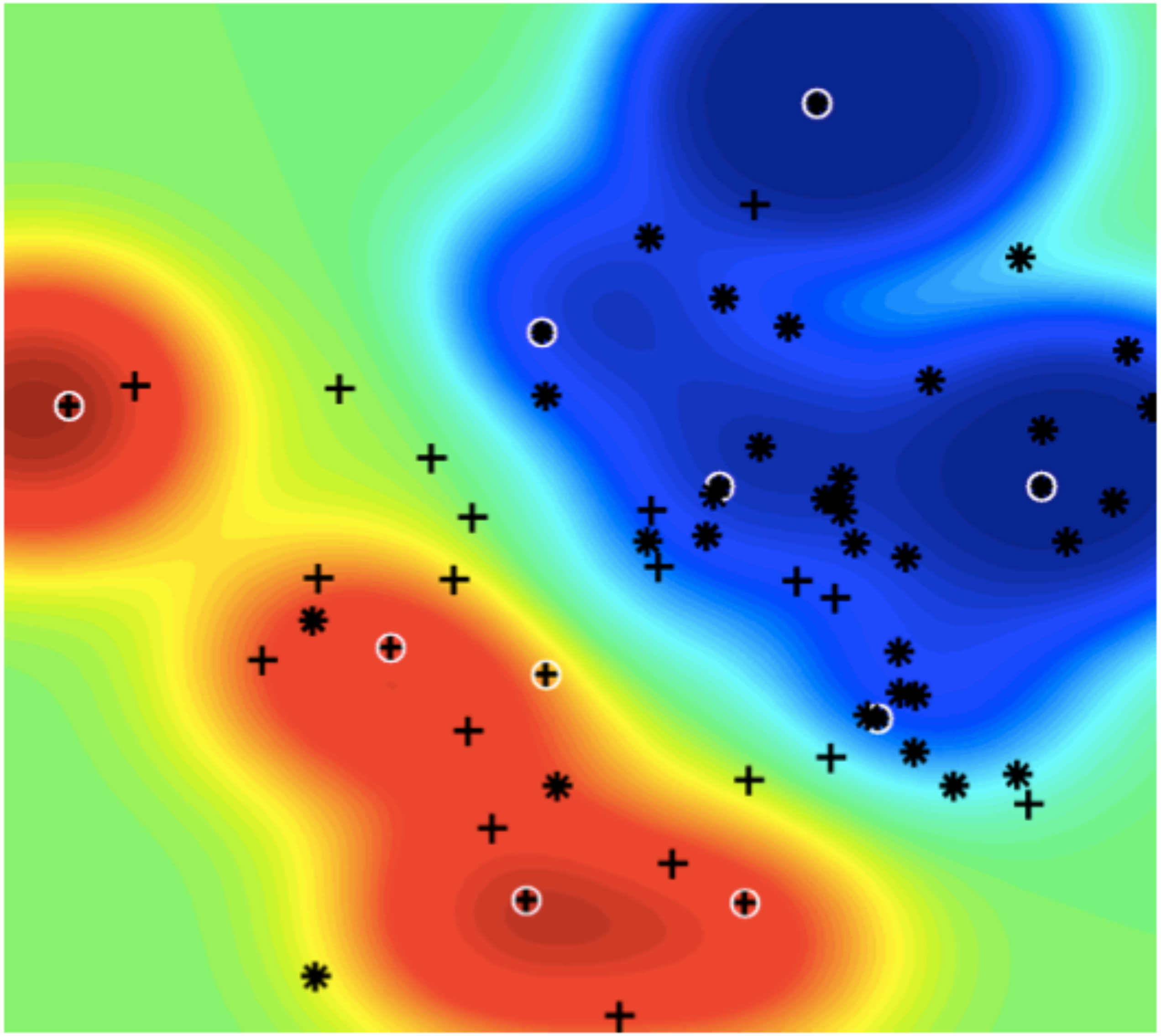
Our Results (logloss) [Liu-**R**-Ziebart '15]



Our Results (error) [Liu-**R**-Ziebart '15]



Predictions



Discussion

Summary

- Showed how to make any ensemble algorithm “feature efficient”.
- Gave an a principled active learning algorithm with impressive empirical performance.

Open Problems

- Better models for trading off error / prediction time?
- Pessimistic active learning applied directly to classification error.

The End

Thank you! Any Questions?