Boosting, Bagging, etc.

Based mostly on notes by Greg Ridgeway

David Madigan

Unstable predictors

We can always assume

$$y = f(\boldsymbol{x}) + \boldsymbol{\varepsilon}\text{, where } \mathrm{E}(\boldsymbol{\varepsilon} \mid \boldsymbol{x}) = 0$$

Assume that we have a way of constructing a predictor, $\hat{\boldsymbol{f}}_D(\boldsymbol{x})$, from a dataset D.

We want to choose the estimator of f that minimizes J, squared loss for example.

$$J(\hat{f}, D) = \mathrm{E}_{y,\boldsymbol{x}}(y - \hat{f}_D(\boldsymbol{x}))^2$$

Bias-variance decomposition

If we could average over all possible datasets, let the average prediction be

$$\bar{f}(\boldsymbol{x}) = \mathrm{E}_D\, \hat{f}_D(\boldsymbol{x})$$

The average prediction error over all datasets that we might see is decomposable

$$\begin{aligned} \mathrm{E}_D\, J(\hat{f}, D) &= \mathrm{E}\,\boldsymbol{\varepsilon}^2 + \mathrm{E}_{\boldsymbol{x}}(f(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}))^2 + \mathrm{E}_{\boldsymbol{x},D}(\hat{f}_D(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}))^2 \\ &= \text{noise} + \text{bias} + \text{variance} \end{aligned}$$

Bias-variance decomposition (cont.)

$$\mathrm{E}_D\, J(\hat{f}, D) = \mathrm{E}\,\boldsymbol{\varepsilon}^2 + \mathrm{E}_{\boldsymbol{x}}(f(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}))^2 + \mathrm{E}_{\boldsymbol{x},D}(\hat{f}_D(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}))^2$$
$$= \text{noise} + \text{bias} + \text{variance}$$

- The noise cannot be reduced.
- The squared-bias term might be reducible
- The variance term is 0 if we use

$$\hat{f}_D(\boldsymbol{x}) = \bar{f}(\boldsymbol{x})$$

But this requires having an infinite number of datasets

Bagging (*B*ootstrap *Agg*regating)

Goal: Variance reduction

Method: Create bootstrap replicates of the dataset and fit a model to each. Average the predictions of each model.

Properties:

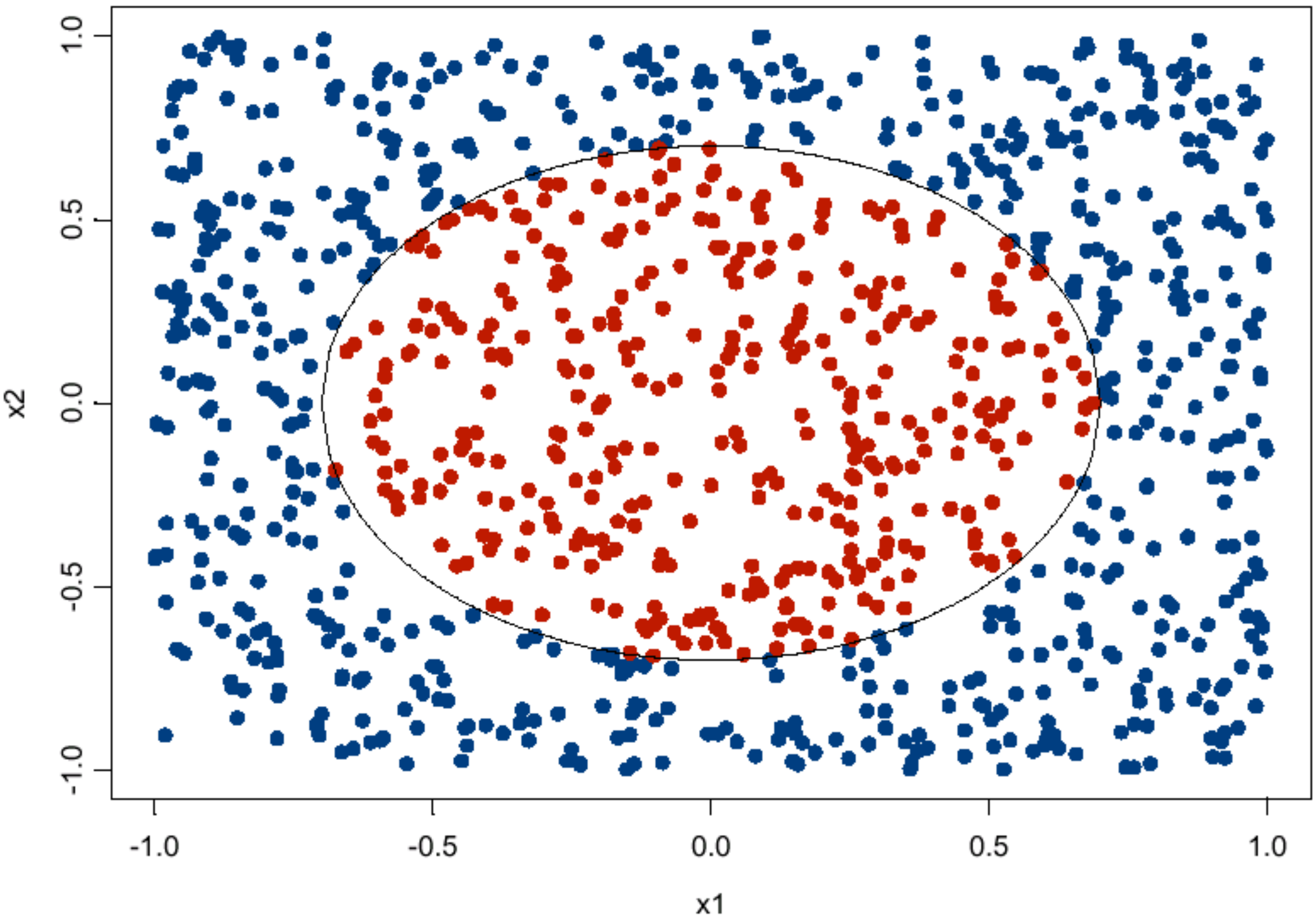
- Stabilizes "unstable" methods
- Easy to implement, parallelizable
- Theory is not fully explained

Bagging algorithm

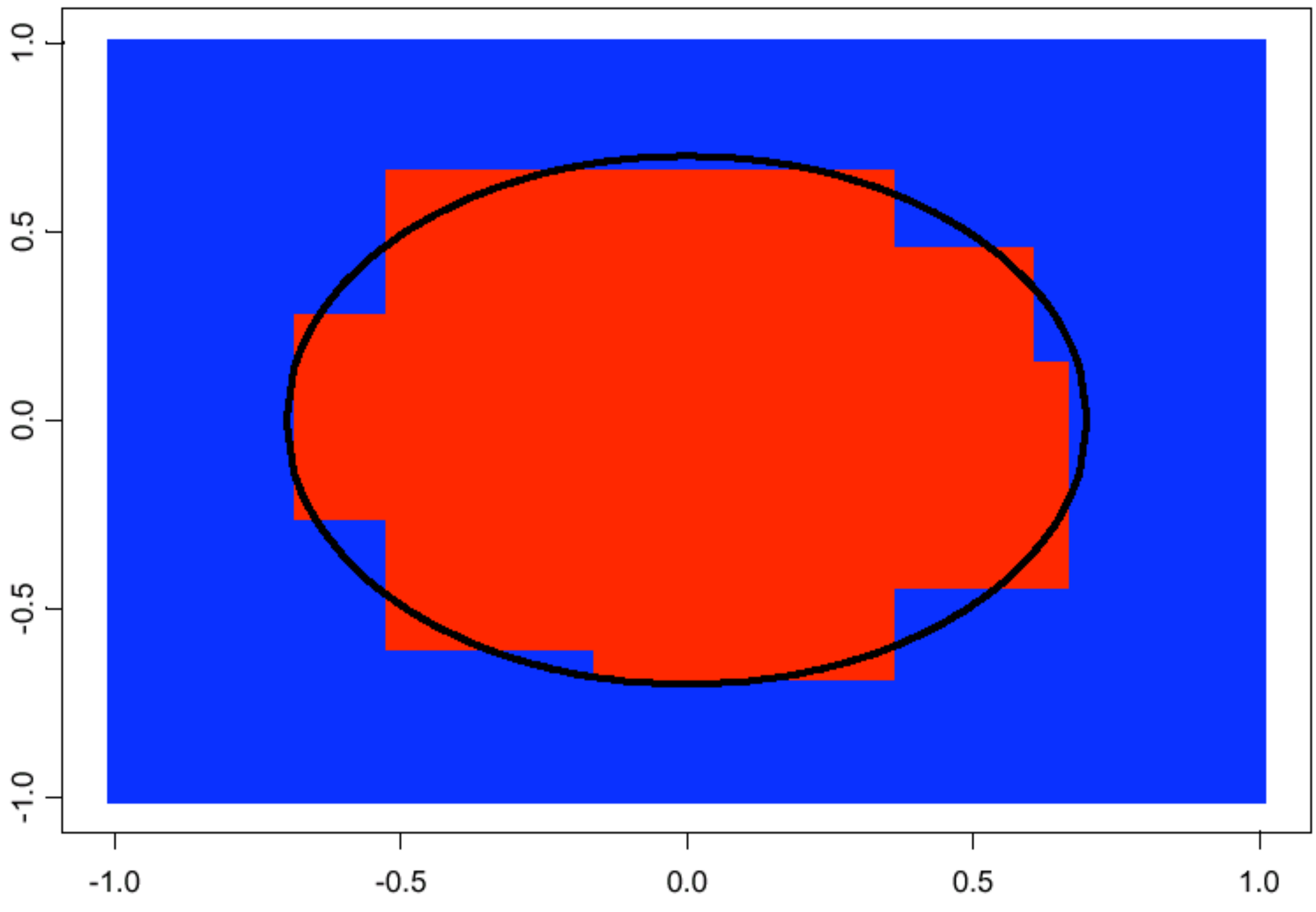
1. Create K bootstrap replicates of the dataset.
2. Fit a model to each of the replicates.
3. Average (or vote) the predictions of the K models.

Bootstrapping simulates the stream of infinite datasets in the bias-variance decomposition.

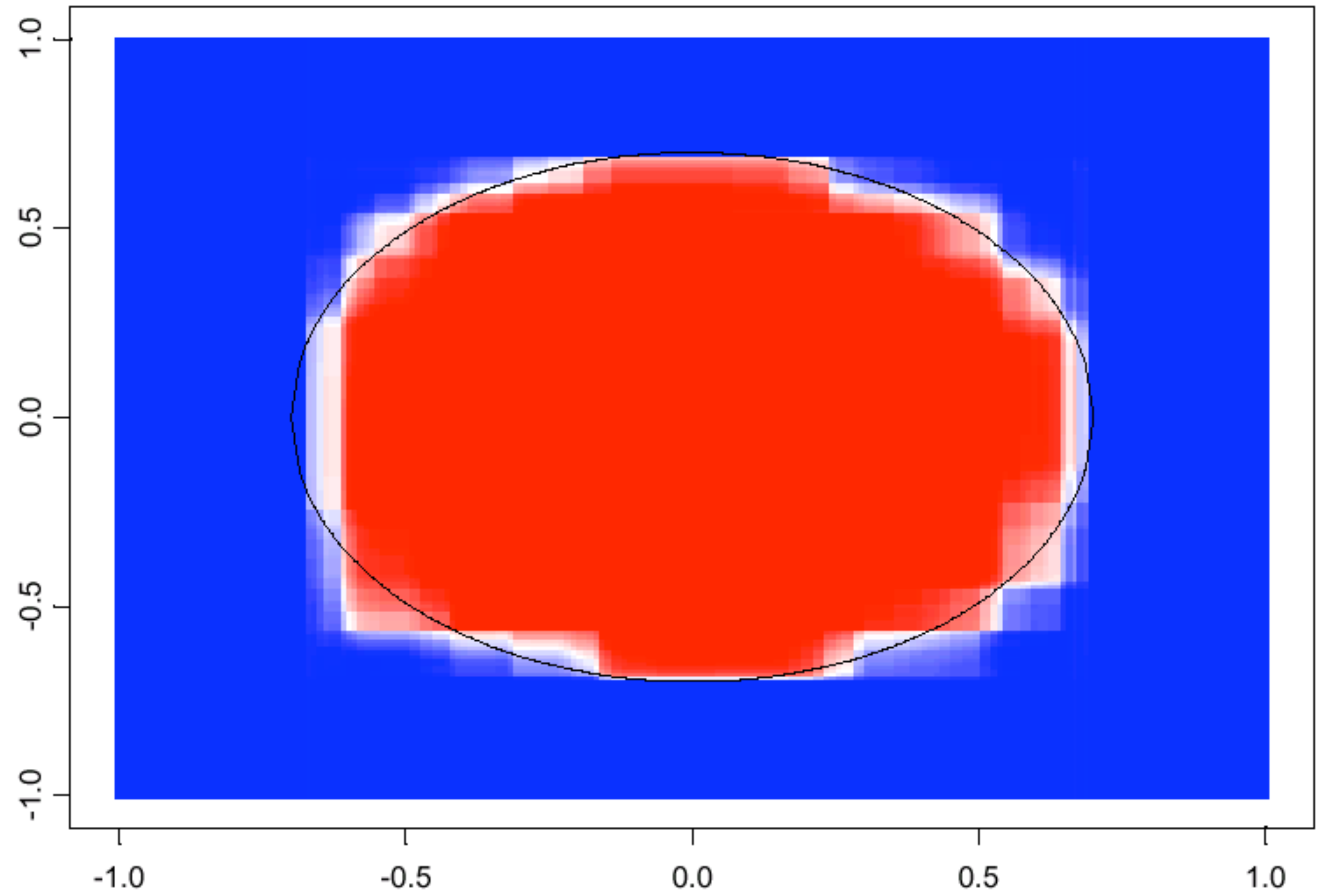
Bagging Example



CART decision boundary

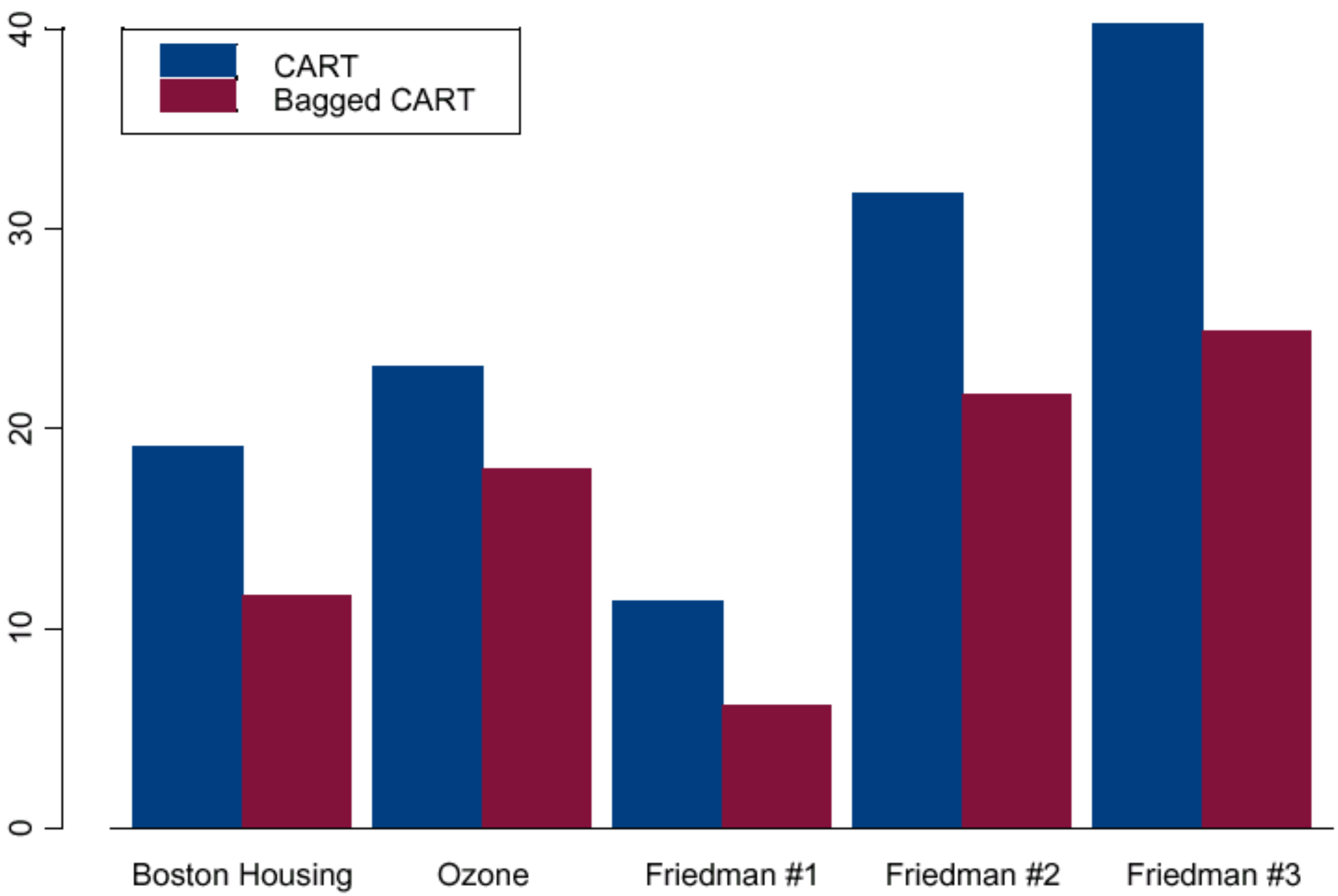


100 bagged trees



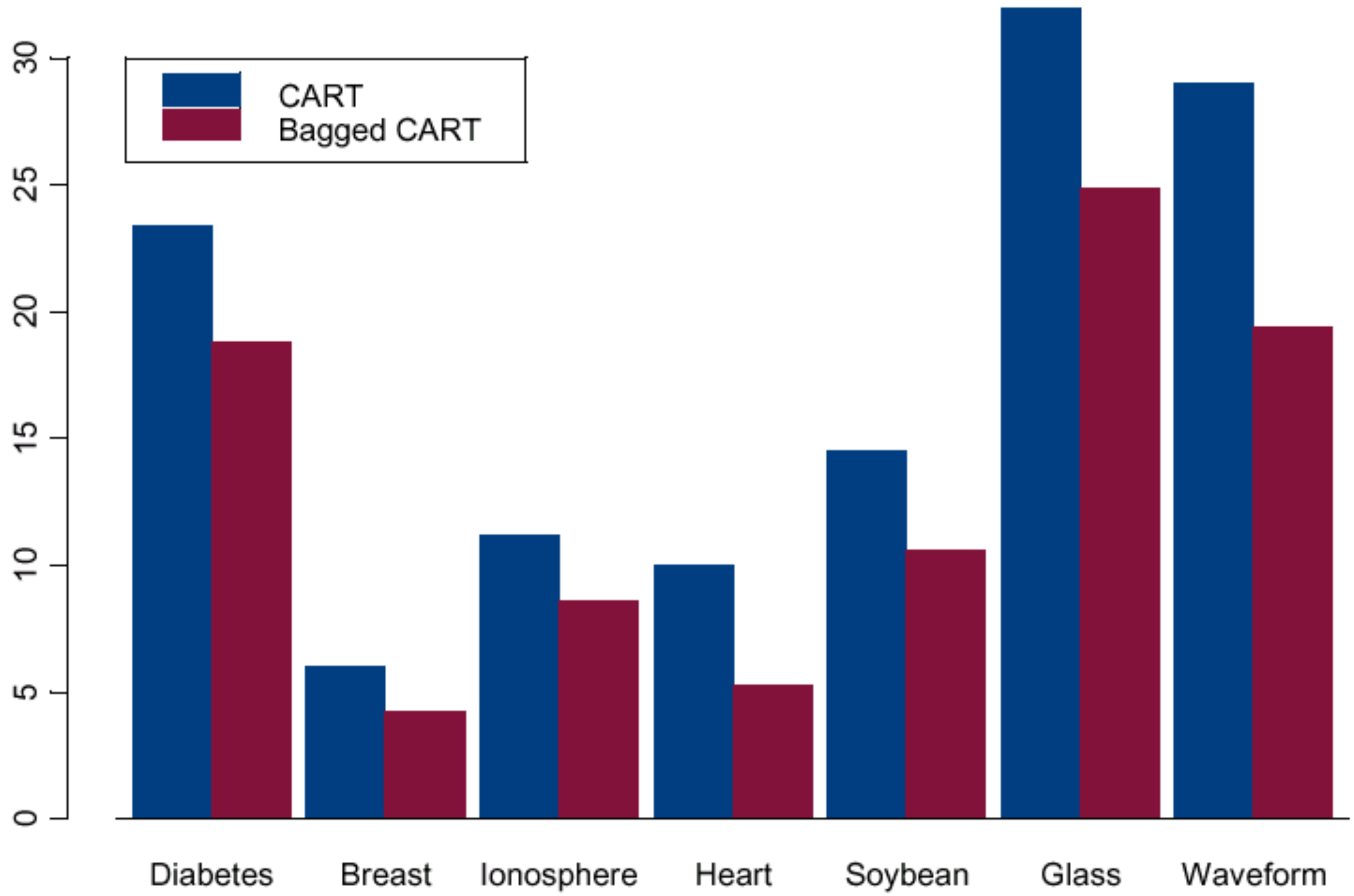
Regression results

Squared error loss



Classification results

Misclassification rates



Random Forests

“ The key to accuracy is low correlation and bias. To keep bias low, trees are grown to maximum depth.

To keep correlation low, the current version uses this randomization.

1) Each tree is grown on a bootstrap sample of the training set.

2) A number m is specified much smaller than the total number of variables M. At each node, m variables are selected at random out of the M, and the split is the best split on these m variables. ”

(see Random Forests , Machine Learning(2001) 45 5-320)

An important feature is that it carries along an internal test set estimate of the prediction error.

For every tree grown, about one-third of the cases are out-of-bag (out of the bootstrap sample). Abbreviated oob.

Put these oob cases down the corresponding tree and get response estimates for them.

For each case n, average or pluralize the response estimates over all time that n was oob to get a test set estimate $\hat{y}_n$ for y_n.

Averaging the loss over all n give the test set estimate of prediction error.

``

Table 3 Test Set Errors (%)

Data Set	Adaboost	Forest-RC		
		Selection	Two Features	One Tree
glass	22.0	24.4	23.5	42.4
breast cancer	3.2	3.1	2.9	5.8
diabetes	26.6	23.0	23.1	32.1
sonar	15.6	13.6	13.8	31.7
vowel	4.1	3.3	3.3	30.4
ionosphere	6.4	5.5	5.7	14.2
vehicle	23.2	23.1	22.8	39.1
German credit	23.5	22.8	23.8	32.6
image	1.6	1.6	1.8	6.0
ecoli	14.8	12.9	12.4	25.3
votes	4.8	4.1	4.0	8.6
liver	30.7	27.3	27.2	40.3
letters	3.4	3.4	4.1	23.8
sat-images	8.8	9.1	10.2	17.3
zip-code	6.2	6.2	7.2	22.7
waveform	17.8	16.0	16.1	33.2
twonorm	4.9	3.8	3.9	20.9
threenorm	18.8	16.8	16.9	34.8
ringnorm	6.9	4.8	4.6	24.6

in R: library(randomForest)

```
> spam7.rf <- randomForest(yesno ~ ., data=spam7, importance=TRUE)
> print(spam7.rf)

Call:
 randomForest(formula = yesno ~ ., data = spam7, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 11.89%
Confusion matrix:
     n    y class.error
n 2642  146  0.05236729
y  401 1412  0.22118036
>
```

```
> tuneRF(x=spam7[,-7], y=spam7$yesno, trace=FALSE)
-0.0582878 0.05
-0.01821494 0.05
  mtry  OOBError
1    1 0.1262769
2    2 0.1193219
4    4 0.1214953
>
```

```
> importance(spam7.rf)
                n         y MeanDecreaseAccuracy MeanDecreaseGini
crl.tot 0.6355890 0.8020131            0.5177274         241.5807
dollar  0.6494899 0.7958824            0.5210265         437.7034
bang    0.6810718 0.8493533            0.5349533         575.5270
money   0.5864884 0.7986897            0.5048771         211.2677
n000    0.6500864 0.4718939            0.5107114         123.9895
make    0.3840864 0.5809521            0.4117850          41.0645
```

Adaptive Bagging

Goal: Bias and variance reduction

Method: Sequentially fit *bagged* models, where each fits the current residuals

Properties:

- Bias and variance reduction
- No tuning parameters

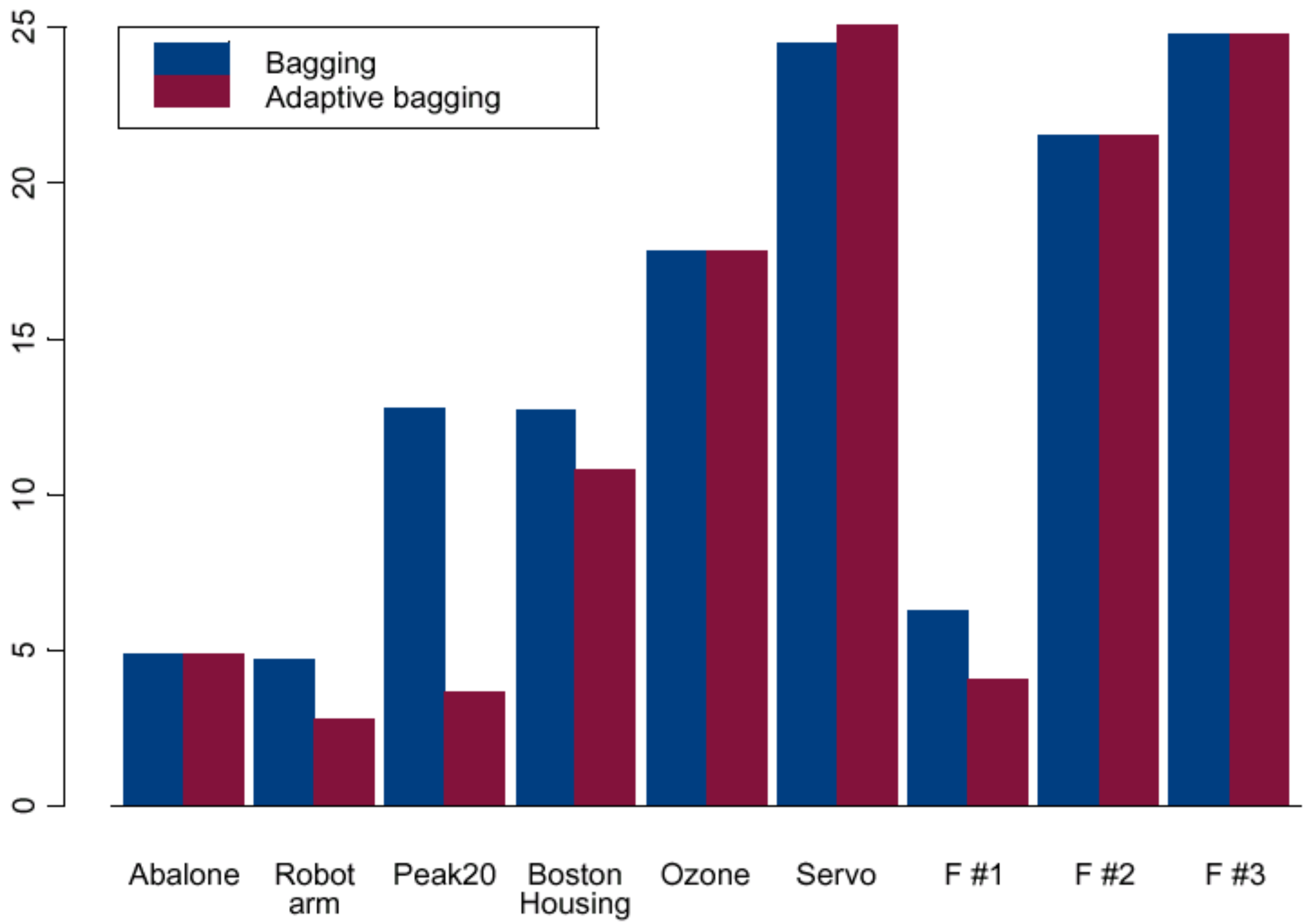
Adaptive bagging algorithm

1. Fit a bagged regressor to the dataset *D*.
2. Predict “out-of-bag” observations.
3. Fit a new bagged regressor to the bias (error) and repeat.

For a new observation, sum the predictions from each stage.

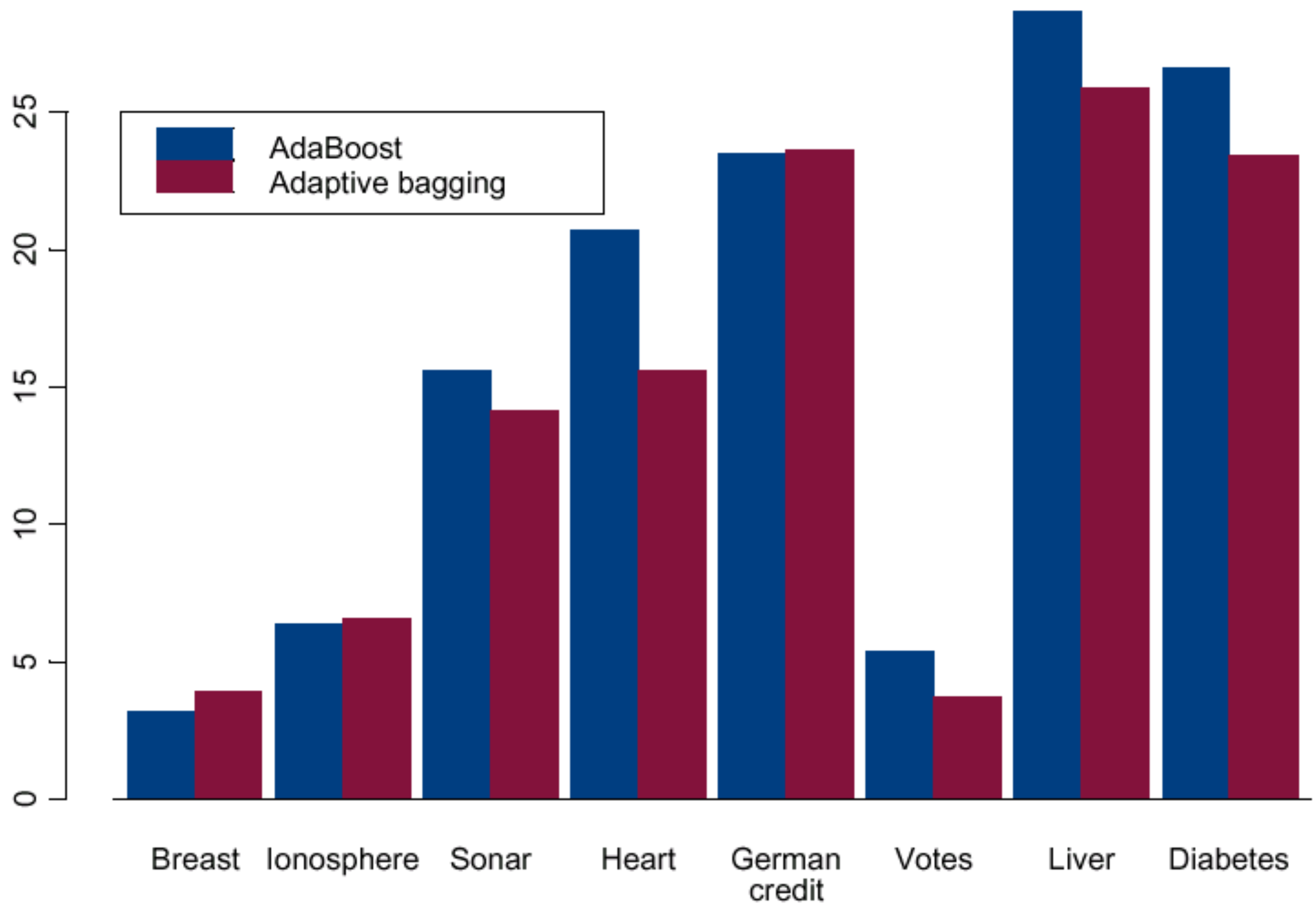
Regression results

Squared error loss



Classification results

Misclassification rates



Bagging References

- Leo Breiman's homepage
  www.stat.berkeley.edu/users/breiman/
- Breiman, L. (1996) "Bagging Predictors," *Machine Learning*, 26:2, 123-140.
- Friedman, J. and P. Hall (1999) "On Bagging and Nonlinear Estimation"
  www.stat.stanford.edu/~jhf

Peter Buhlmann and Bin Yu. Explaining bagging. Can be downloaded from http://stat.ethz.ch/~buhlmann/bibliog.html, September 2000.

J.H. Friedman and O. Hall. On bagging and nonlinear estimation. Can be downloaded from http://www-stat.stanford.edu/~jhf/#reports, May 2000.

Andreas Buja's home page:

``The Effect of Bagging on Variance, Bias and Mean Squared Error"
A. Buja, W. Stuetzle.
Bootstrap aggregation ("bagging") is a device for reducing the variance of learning algorithms. We give a complete second-order analysis of the effect of bagging on finite sums of U-statistics.

``Smoothing Effects of Bagging"
A. Buja, W. Stuetzle.
An short note on bagging. It relates the von Mises expansion of a bagged statistical functional to the Efron-Stein ANOVA expansion of the unbagged functional to show that the bagged functional is always smooth.

Boosting

Goal: Improve misclassification rates

Method: Sequentially fit models, each more heavily weighting those observations poorly predicted by the previous model

Properties:

- Bias and variance reduction
- Easy to implement
- Theory is not fully (but almost) explained

Generic boosting algorithm

Equally weight the observations $(y,\boldsymbol{x})_i$

For t in $1,\ldots,T$

- Using the weights, fit a classifier $f_t(\boldsymbol{x}) \to y$
- Upweight the poorly predicted observations
- Downweight the well-predicted observations

Merge $f_1,\ldots,f_T$ to form the boosted classifier

Real AdaBoost

Schapire & Singer 1998

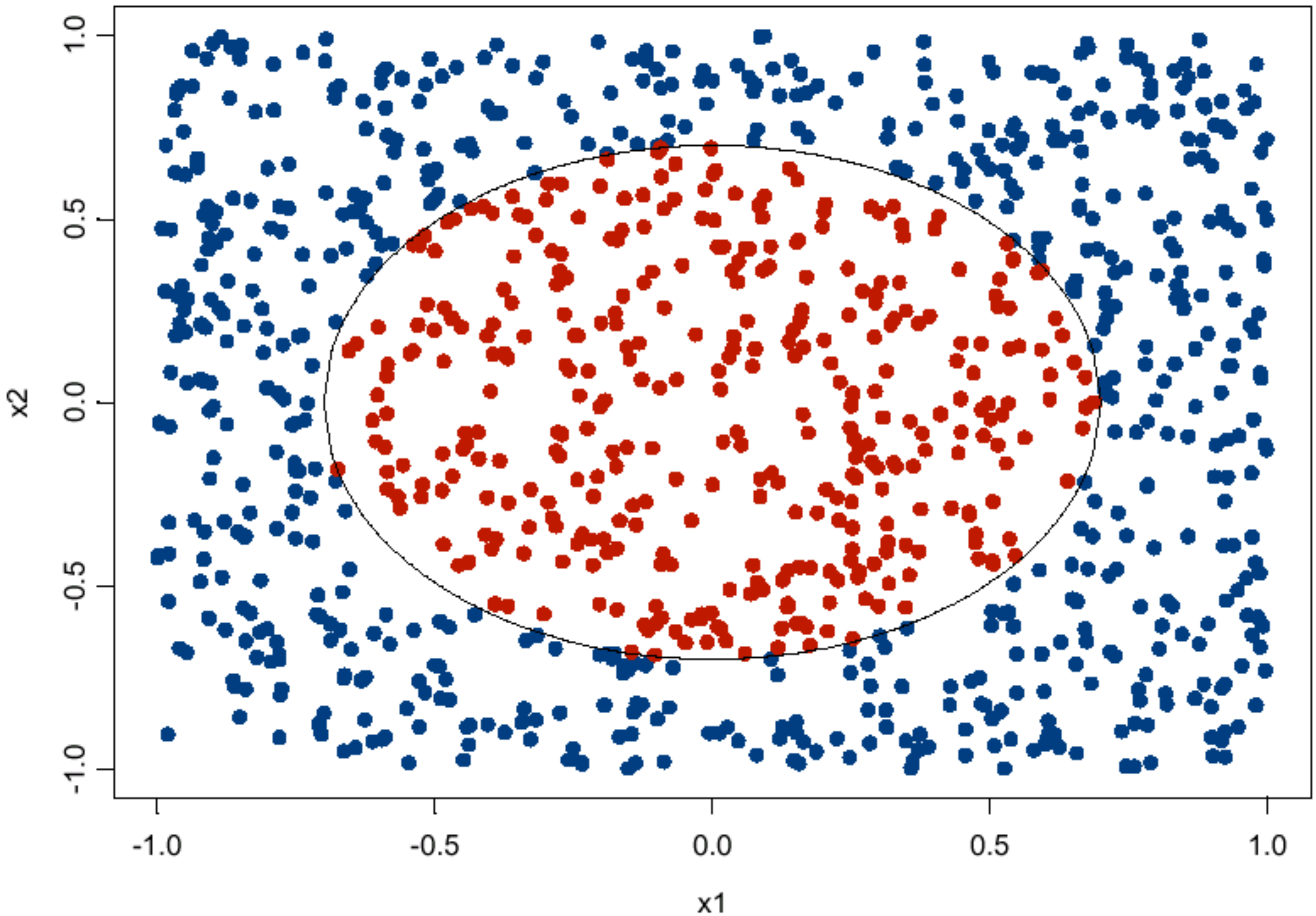
$y_i \in \{-1,1\}$, $w_i = 1/N$

For t in $1,\ldots,T$ do

1. Estimate $\mathrm{P}_w(y = 1|\boldsymbol{x})$.
2. Set $f_t(\boldsymbol{x}) = \frac{1}{2}\log\frac{\hat{\mathrm{P}}_w(y=1\,|\,\boldsymbol{x})}{\hat{\mathrm{P}}_w(y=-1\,|\,\boldsymbol{x})}$
3. $w_i \leftarrow w_i \exp\left(-y_i f_t(\boldsymbol{x}_i)\right)$ and renormalize

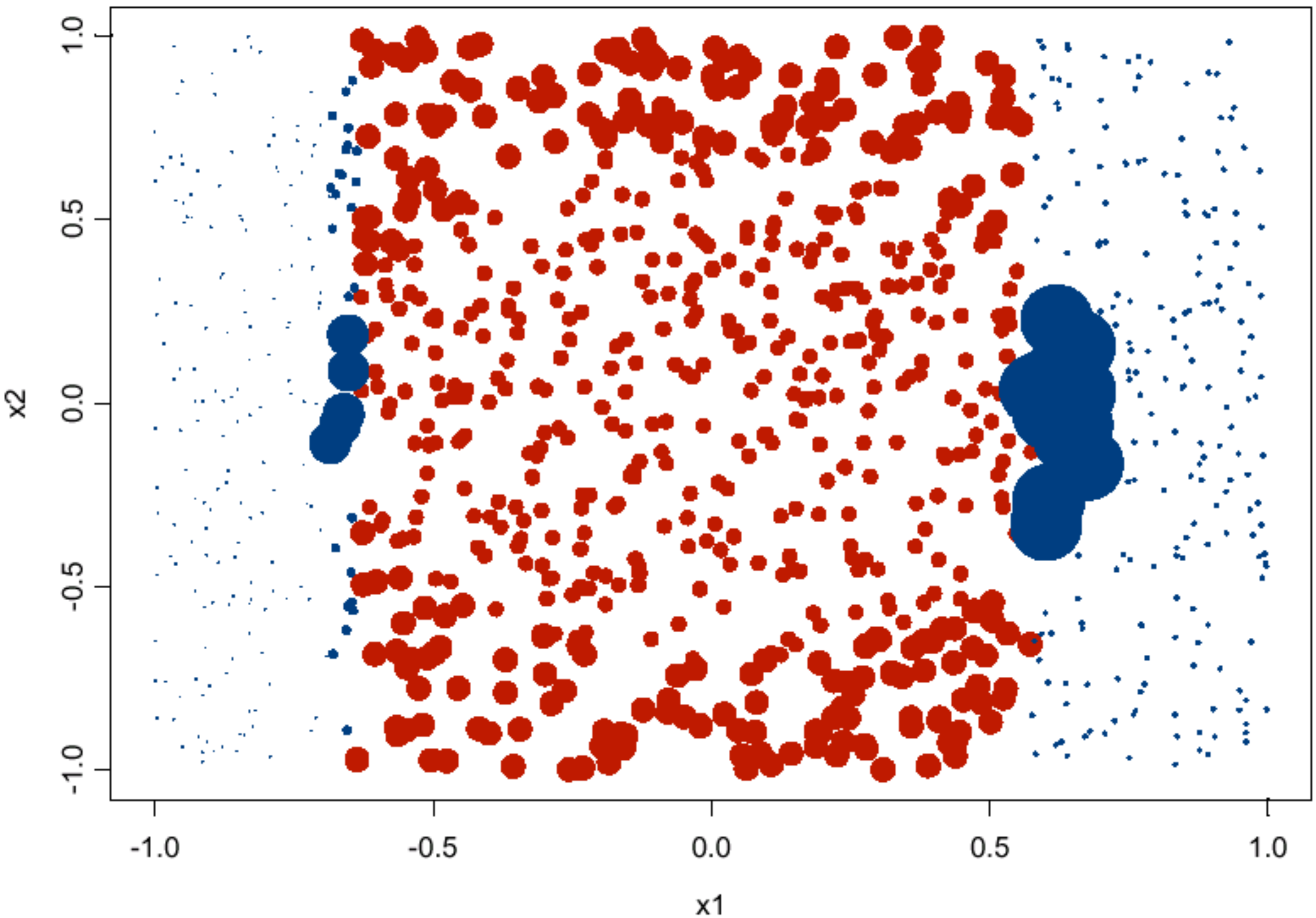
Output the classifier $\boldsymbol{F}(\boldsymbol{x}) = \mathrm{sign}\left(\sum f_t(\boldsymbol{x})\right)$

Boosting Example

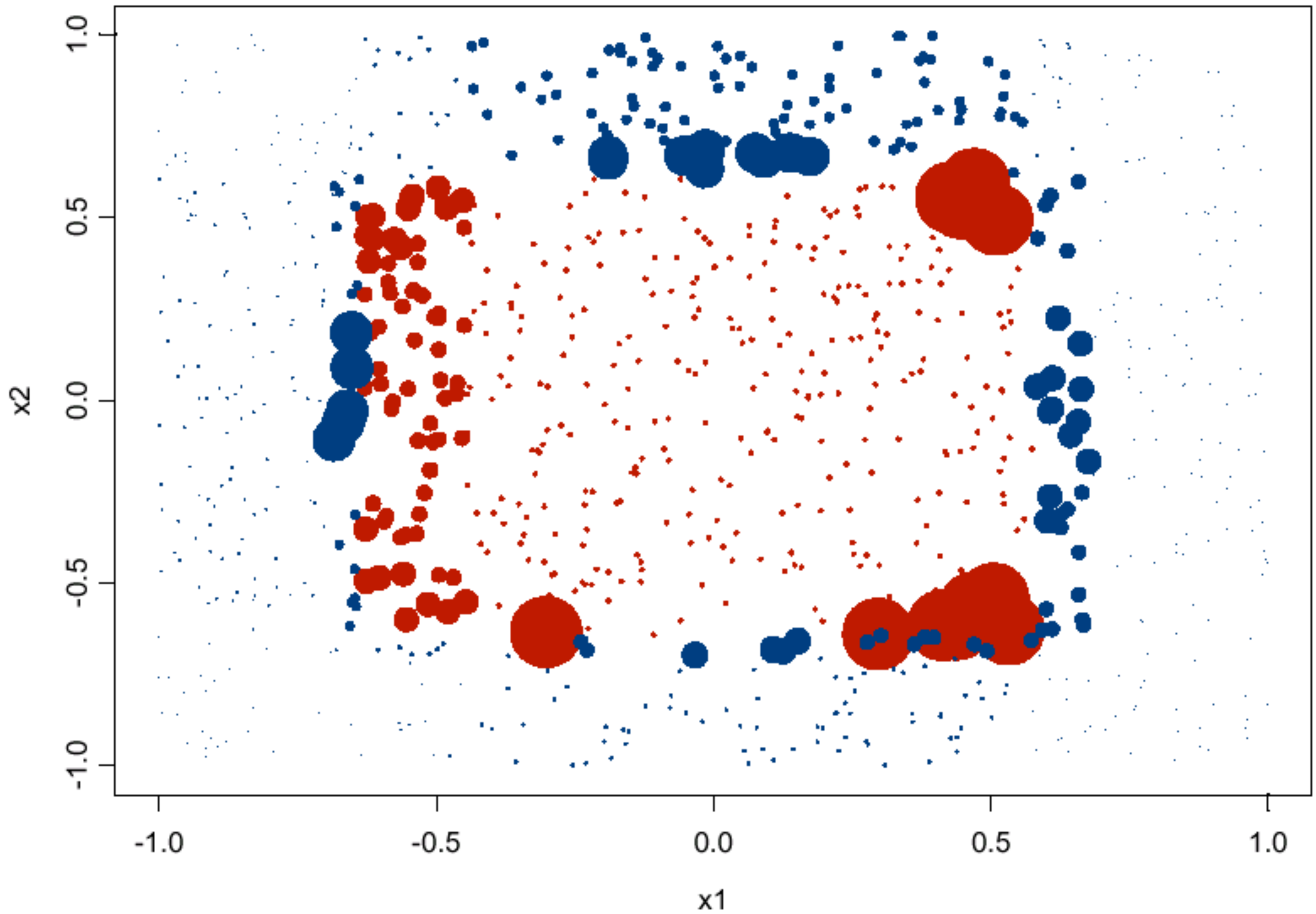


After one iteration

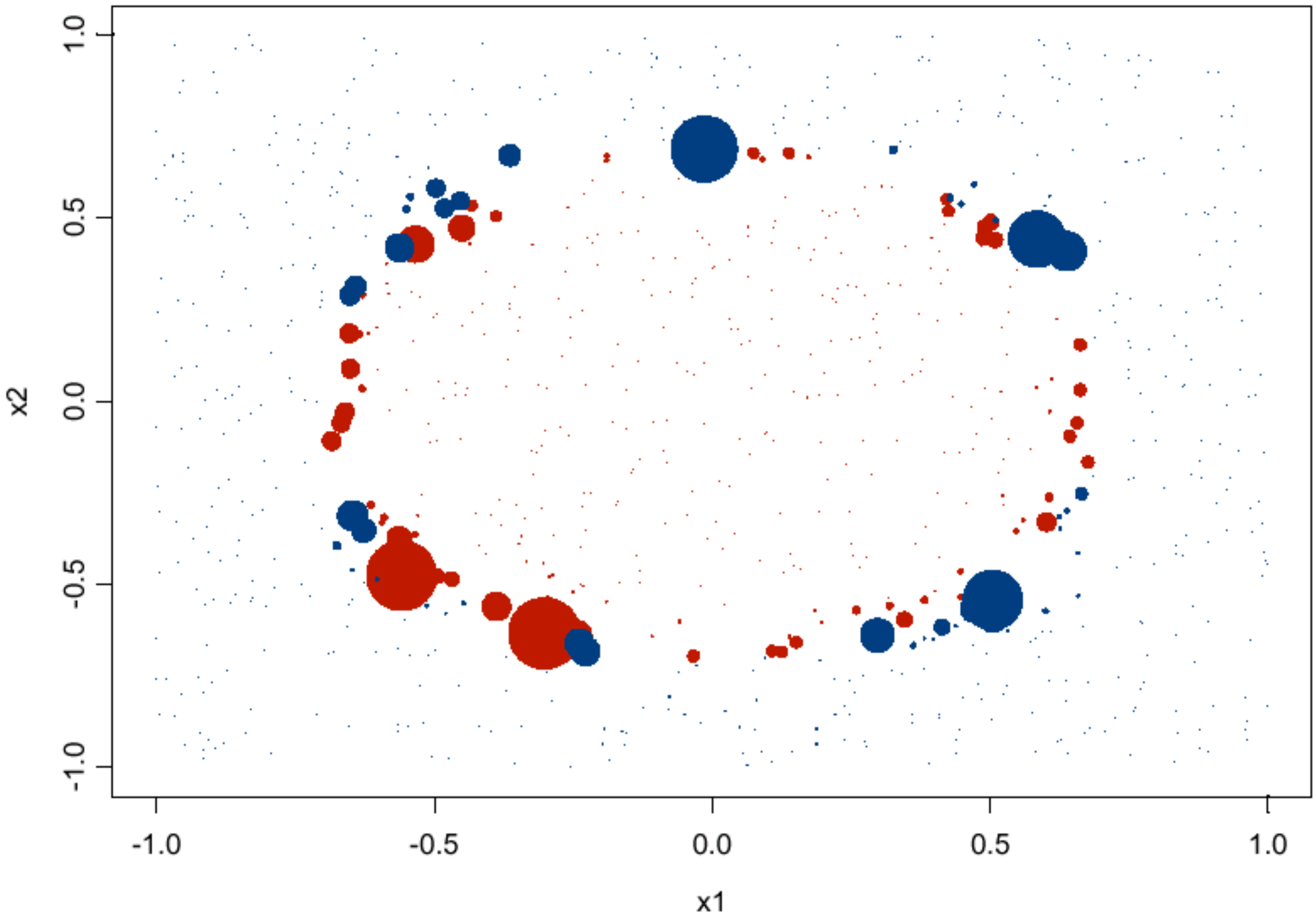
CART splits, larger points have great weight



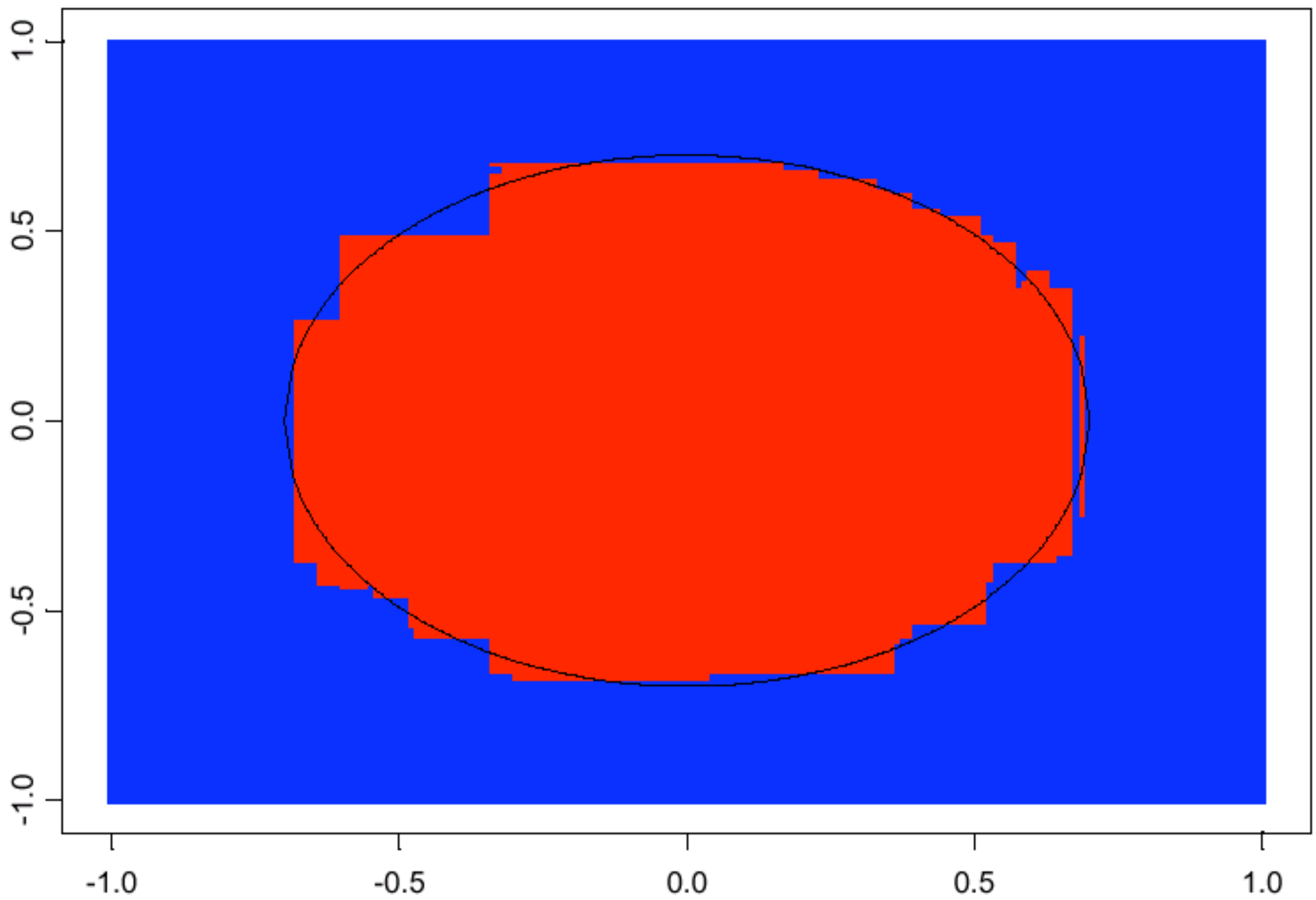
After 3 iterations



After 20 iterations



Decision boundary after 100 iterations



Boosting as optimization

- Friedman, Hastie, Tibshirani [1998] - AdaBoost is an optimization method for finding a classifier.
- Let $y \in \{-1,1\}$, $F(x) \in (-\infty,\infty)$

$$J(F) = E\left(e^{-yF(x)} \mid x\right)$$

Criterion

- $E(e^{-yF(x)})$ bounds the misclassification rate.

$$I(yF(x) < 0) < e^{-yF(x)}$$

- The minimizer of $E(e^{-yF(x)})$ coincides with the maximizer of the expected Bernoulli likelihood.

$$J(F) = \mathrm{E}\ell(F) = \mathrm{E}\left[y^* F(\mathbf{x}) - \log\left(1 + e^{F(\mathbf{x})}\right) |\mathbf{x}\right]$$

$$y^* = \frac{1}{2}(1+y) \in \{0, 1\}$$

Optimization step

$$J(F+f)=E\left(e^{-y\left(F(x)+f(x)\right)}\mid x\right)$$

- Select f to minimize J…

$$F^{(t+1)} \leftarrow F^{(t)}+\tfrac{1}{2}\log\frac{E_w[I(y=1)\mid x]}{1-E_w[I(y=1)\mid x]}$$

$$w(x,y)=e^{-yF^{(t)}(x)}$$

Let $J(F) = E[e^{-yF(x)}]$. Suppose we have a current estimate $F(x)$ and seek an improved estimate $F(x) + cf(x)$. For fixed c (and x), we expand $J(F(x) + cf(x))$ to second order about $f(x) = 0$

$$\begin{aligned} J(F + cf) &= E[e^{-y(F(x)+cf(x))}] \\ &\approx E[e^{-yF(x)}(1 - ycf(x) + c^2y^2f(x)^2/2)] \\ &= E[e^{-yF(x)}(1 - ycf(x) + c^2/2)] \end{aligned}$$

since y^2=1 and $f(x)^2 = 1$. Minimizing pointwise with respect to $f(x) \in \{-1, 1\}$, we write

$$f(x) = \arg\min_f E_w(1 - ycf(x) + c^2/2|x) \tag{16}$$

Here the notation $E_w(\cdot|x)$ refers to a *weighted conditional expectation*, where $w = w(x, y) = e^{-yF(x)}$, and

$$E_w[g(x, y)|x] \stackrel{\text{def}}{=} \frac{E[w(x, y)g(x, y)|x]}{E[w(x, y)|x]}.$$

For $c > 0$, minimizing (16) is equivalent to maximizing

$$E_w[yf(x)] \tag{17}$$

The solution is

$$f(x) = \begin{cases} 1 & \text{if } E_w(y|x) = P_w(y = 1|x) - P_w(y = -1|x) > 0 \\ -1 & \text{otherwise} \end{cases} \tag{18}$$

LogitBoost

Friedman, Hastie, Tibshirani [1998]

- Logistic regression

$$y = \begin{cases} 1 & \text{with probability } p(x) \\ 0 & \text{with probability } 1 - p(x) \end{cases}$$

$$p(x) = \frac{1}{1 + e^{-F(x)}}$$

- Expected log-likelihood of a regressor, $F(x)$

$$\mathrm{E}\,\ell(F) = \mathrm{E}\left(yF(x) - \log(1 + e^{F(x)}) \mid x\right)$$

Newton steps

$$J(F+f) = E\left(y(F(x)+f(x)) - \log(1+e^{F(x)+f(x)}) \mid x\right)$$

- Iterate to optimize expected log-likelihood.

$$F^{(t+1)}(x) \leftarrow F^{(t)}(x) - \frac{\frac{\partial}{\partial f} J(F^{(t)}+f)\Big|_{f=0}}{\frac{\partial^2}{\partial f^2} J(F^{(t)}+f)\Big|_{f=0}}$$

LogitBoost, continued

- Newton steps for Bernoulli likelihood

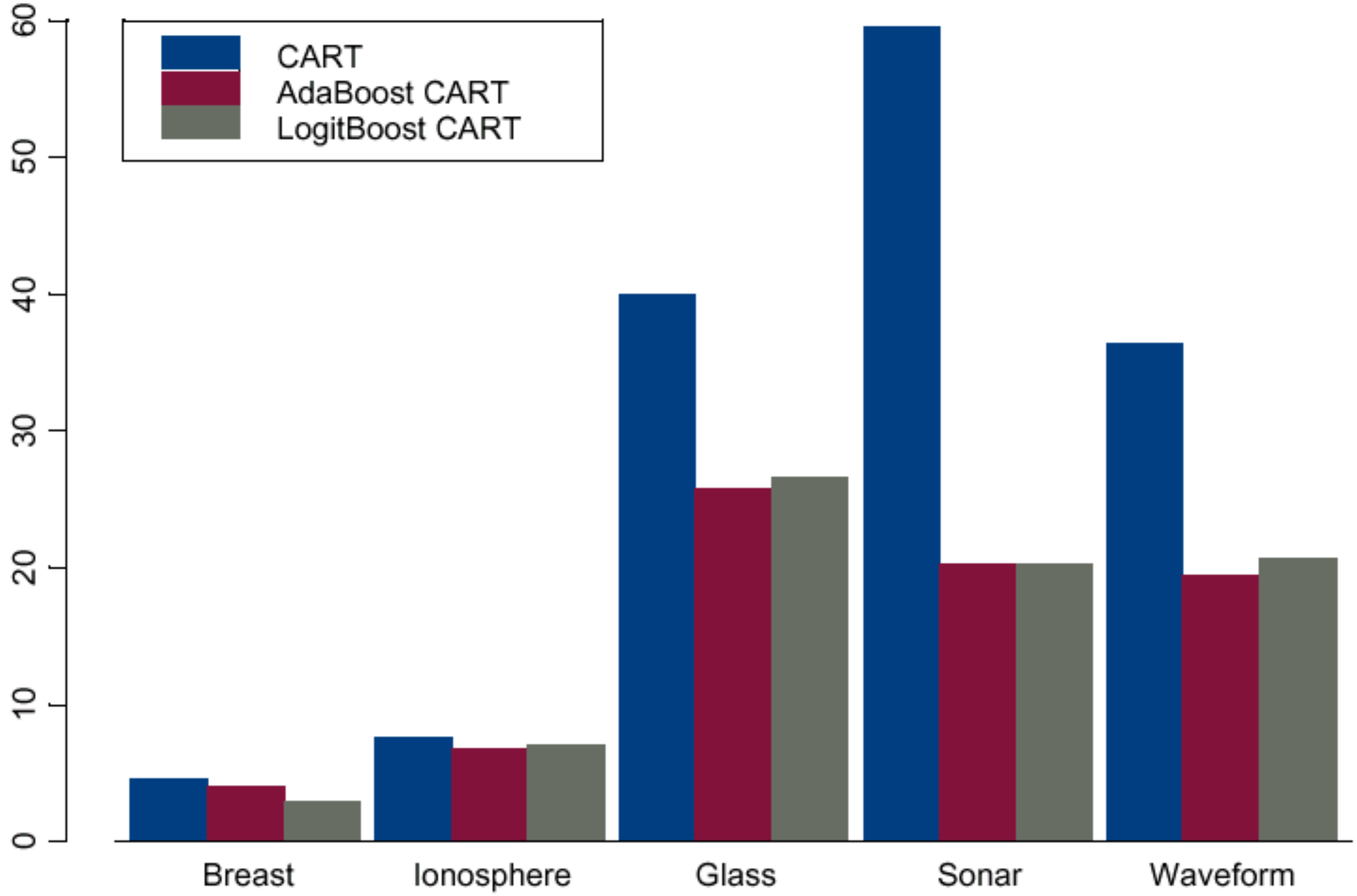
$$F(x) \leftarrow F(x) + E_w\left(\frac{y - p(x)}{p(x)(1 - p(x))}\middle| x\right)$$

$$w(x) = p(x)(1 - p(x))$$

- In practice the $E_w(\bullet|x)$ can be any regressor - trees, smoothers, etc.
- Trees are adaptive and work well for high dimensional data.

Misclassification rates

Friedman, Hastie, Tibshirani [1998]



Boosting References

- Rob Schapire’s homepage

http://www.cs.princeton.edu/~schapire/boost.html

- Freund, Y. and R. Schapire (1996). “Experiments with a new boosting algorithm,” Machine Learning: Proceedings of the 13th International Conference, 148-156.

- Jerry Friedman’s homepage

www.stat.stanford.edu/~jhf

- Friedman, J., T. Hastie, R. Tibshirani (1998). “Additive Logistic Regression: a statistical view of boosting,” Technical report, Statistics Department, Stanford University.