Calibrating AdaBoost for asymmetric learning

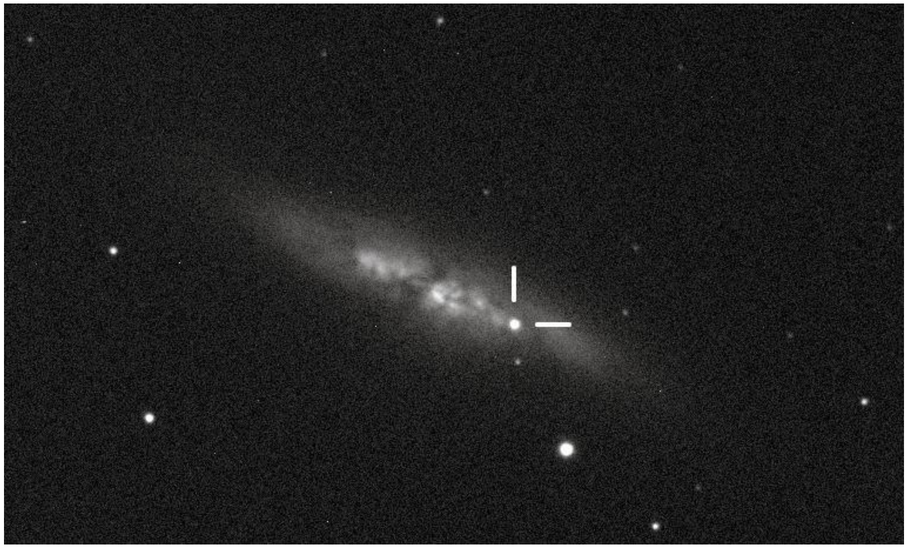
Nikos Nikolaou, Gavin Brown

MANCHESTER 1824

The University of Manchester

Asymmetric Learning

Cost-sensitive
False Positives & False Negatives
have different costs



Imbalanced classes
Positives & Negatives
have different priors

...or both!

Some Conventions

- **Binary** classification: $y \in \{-1, 1\} \equiv \{Neg, Pos\}$

- Can model asymmetry using **skew ratio** c:

$$c = \frac{importance\ of\ a\ Pos}{importance\ of\ a\ Neg}$$

  – Imbalanced classes ($importance = rarity$):

$$c = \frac{p(Neg)}{p(Pos)}$$

  – Cost-sensitive ($importance = cost\ of\ misclassifying$)

$$c = \frac{C_{FN}}{C_{FP}}$$

AdaBoost

$$\alpha_t = \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

$$\epsilon_t = \sum_{i:h_t(\mathbf{x}_i)\neq y_i} D_i^t$$

Assign a confidence score to each weak learner

Can it handle asymmetric problems?

$$D_i^{t+1} = e^{-y_i h_t(\mathbf{x}_i)\alpha_t} D_i^t$$

Update examples' weights

$$H(\mathbf{x}') = sign\left[\sum_{t=1}^{M} \alpha_t h_t(\mathbf{x}')\right]$$

Confidence weighted majority vote

Asymmetric Boosting Variants

$$\alpha_t = \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

$$\epsilon_t = \sum_{i:h_t(\mathbf{x}_i)\neq y_i} D_i^t$$

Assign a confidence score to each weak learner

(Fan et al., 1999)
(Cohen & Singer, 1999)
(Ting, 2000)
(Joshi et al., 2001)
(Sun et al., 2005; 2007)
(Masnadi-Shirazi & Vasconcelos , 2007; 2011)

$$D_i^{t+1} = e^{-y_i h_t(\mathbf{x}_i)\alpha_t} D_i^t$$

Update examples' weights

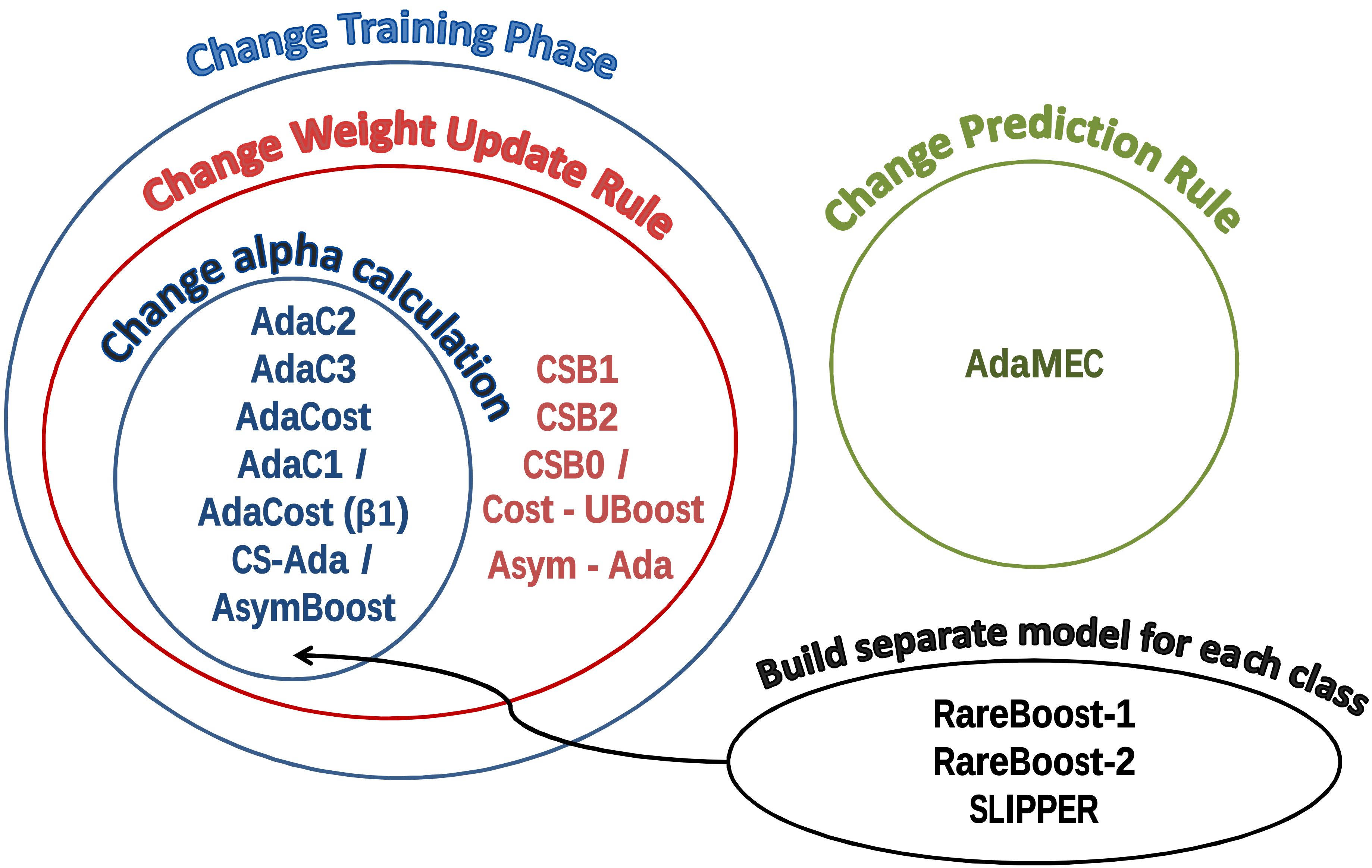
(Ting & Zheng, 1998)
(Ting, 2000)
(Viola & Jones, 2001; 2002)

$$H(\mathbf{x}') = sign\left[\sum_{t=1}^{M}\alpha_t h_t(\mathbf{x}')\right]$$

Confidence weighted majority vote

(Ting, 2000)

Asymmetric Boosting Variants



Issues with modifying training phase

- No **theoretical guarantees** of original AdaBoost
  - e.g. bounds on generalization error, convergence

- Most heuristic, **no decision-theoretic** motivation
  - ad-hoc changes, not apparent what they achieve

- Need to **retrain** if skew ratio changes

- Require **extra hyperparameters** to be set via CV

Issues with modifying prediction rule

- *AdaMEC* changes prediction rule from **weighted majority** vote

$$H(\mathbf{x}') = sign\left[\sum_{t=1}^{M} \alpha_t h_t(\mathbf{x}')\right]$$

to **minimum expected cost** criterion

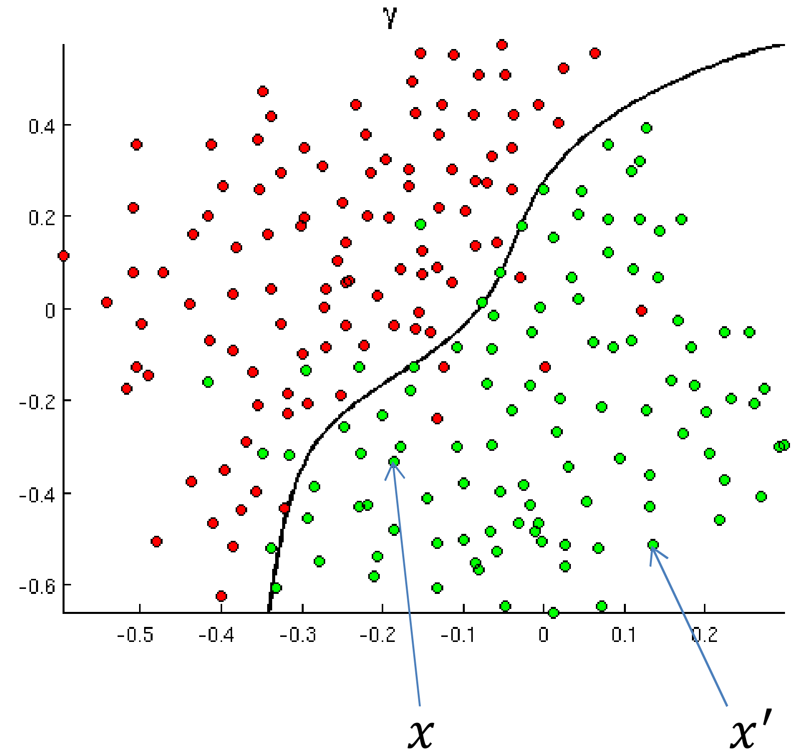
$$H(\mathbf{x}') = sign\left[\sum_{y\in\{-1,1\}} c(y)\sum_{t=1}^{M} \alpha_t h_t(\mathbf{x}')\right] \quad , \quad c(y) = \begin{cases} c, & \text{if } y_i = 1 \\ 1, & \text{if } y_i = -1 \end{cases}$$

- Problem: **incorrectly assumes** **scores** are reliable **probability estimates**...

- ...but can correct this via **calibration**

Things classifiers do…

- **Classify** examples
  - Is x positive?
- **Rank** examples
  - Is x 'more positive' than x'?
- Output a **score** for each example
  - 'How positive' is x?
- Output a **probability estimate** for each example
  - What is the (estimated) probability that x is positive?



Why estimate probabilities?

- Need **probabilities** when a **cost-sensitive decision** needs to be made; scores won't cut it

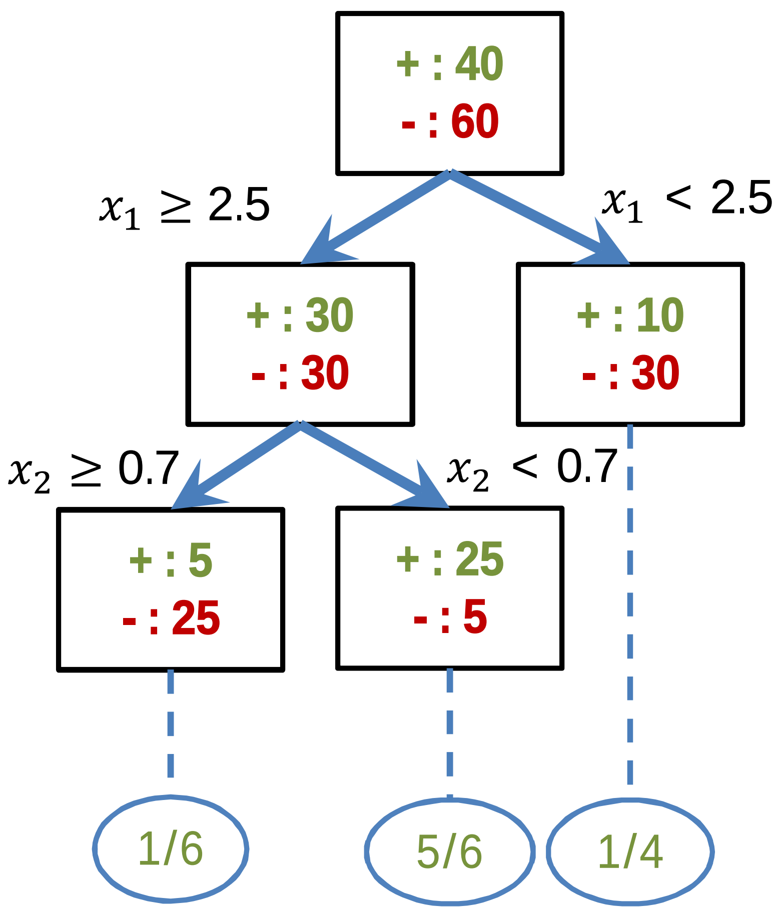
- Will assign to class that minimizes **expected** cost i.e. assign to $y = 1$ (Pos) only if:

$$\hat{p}(y = 1|\mathbf{x}')c > \hat{p}(y = -1|\mathbf{x}') \iff \hat{p}(y = 1|\mathbf{x}') > \frac{1}{1+c}$$

(We set $C_{FP} = 1$, thus $c = C_{FN}$)

Probability estimation is not easy

Most classifiers don't produce probability estimates **directly** but we get them via scores, e.g. decision trees:



Tree as constructed on training set

Score of test example that falls on leaf; Should we take this as $\hat{p}(+|x)$?

Even 'probabilistic' classifiers can fail to produce **reliable** probability estimates (e.g. Naïve Bayes)

Calibration

- $s(x) \in [0,1]$: score assigned by classifier to example x ('how positive' x is)

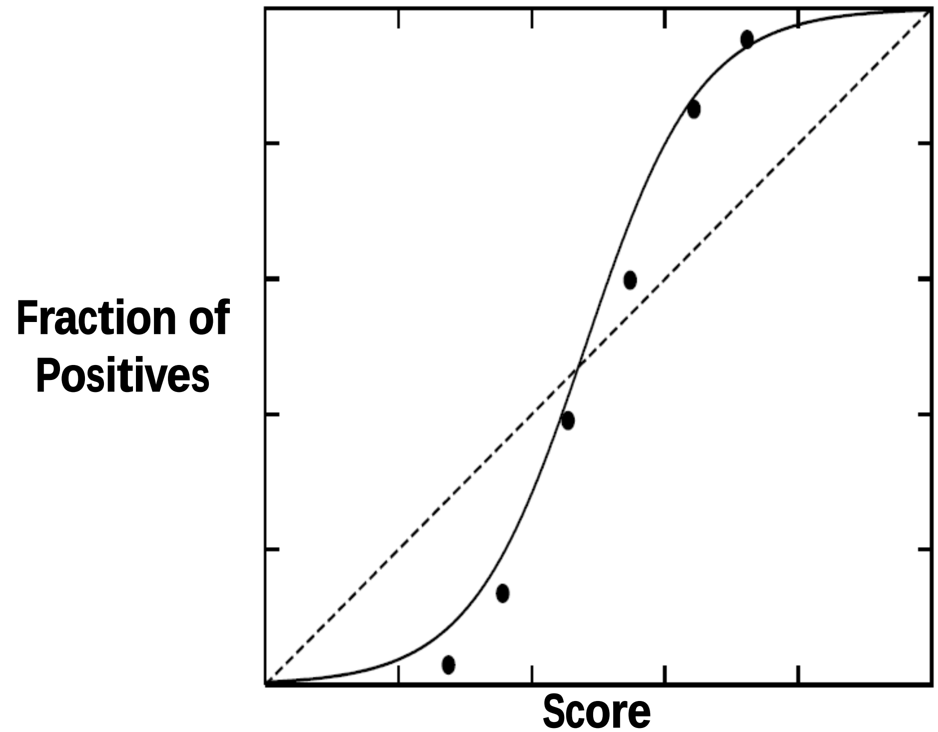
- A classifier is **calibrated** if
  $$\hat{p}(y = 1|x) \rightarrow s(x), \text{ as } N \rightarrow \infty$$

- Intuitively: consider all examples with $s(x) = 0.7$;
  70% of these examples **should** be positives

- Calibration **can only improve** classification

Probability estimates of AdaBoost

Score for Boosting: $s(\mathbf{x}') = \frac{\sum_{t=1}^{M} \alpha_t \frac{h_t(\mathbf{x}')+1}{2}}{\sum_{t=1}^{M} \alpha_t}$



Boosted trees / stumps: **sigmoid distortion**; scores pushed more towards 0 or 1 as num. of boosting rounds increases

(Niculescu-Mizil & Caruana, 2006)

How to calibrate AdaBoost

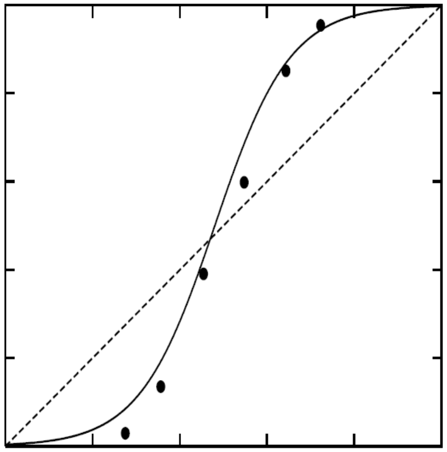
- **Logistic Correction**
- **Isotonic Regression**
- **Platt Scaling**
  - – Suitable if distortion is sigmoid (base-learner dependent)
  - – Best results when data limited

Platt Scaling

- Find A, B for $\hat{p}(y = 1|\mathrm{x}) = \frac{1}{1+e^{A\,s(x)\,+\,B}}$, s. t. likelihood of data is maximized

- **Separate sets** for train & calibration

- Motivation: undo sigmoid distortion observed in boosted trees



Calibrating AdaBoost for asymmetric learning

On training set:

- Train AdaBoost ensemble H_M

↓

On validation set:

- Calculate score $s(\mathbf{x}) = \frac{\sum_{t=1}^{M} \alpha_t \frac{h_t(\mathbf{x})+1}{2}}{\sum_{t=1}^{M} \alpha_t} \in [0,1]$ of each example $\mathbf{x}$ under ensemble H_M
- Find A, B s. t. the likelihood of the data under model $\hat{p}(y=1|\mathbf{x}) = \frac{1}{1+e^{As(\mathbf{x})+B}}$ is maximized

↓

On test set:

- Calculate score $s(\mathbf{x})$, $\forall$ example $\mathbf{x}$ under H_M
- Apply transformation $\hat{p}(y=1|\mathbf{x}) = \frac{1}{1+e^{As(\mathbf{x})+B}}$ to the scores $s(\mathbf{x})$ to get probability estimates
- Predict class $H_M(\mathbf{x}) = sign[\hat{p}(y=1|\mathbf{x}) - \frac{1}{1+c}]$

Experimental Design

- AdaC2 vs. CSB2 vs. AdaMEC vs. Calibrated AdaBoost

  75% Tr / 25% Te 50% Tr / 25% Cal / 25% Te

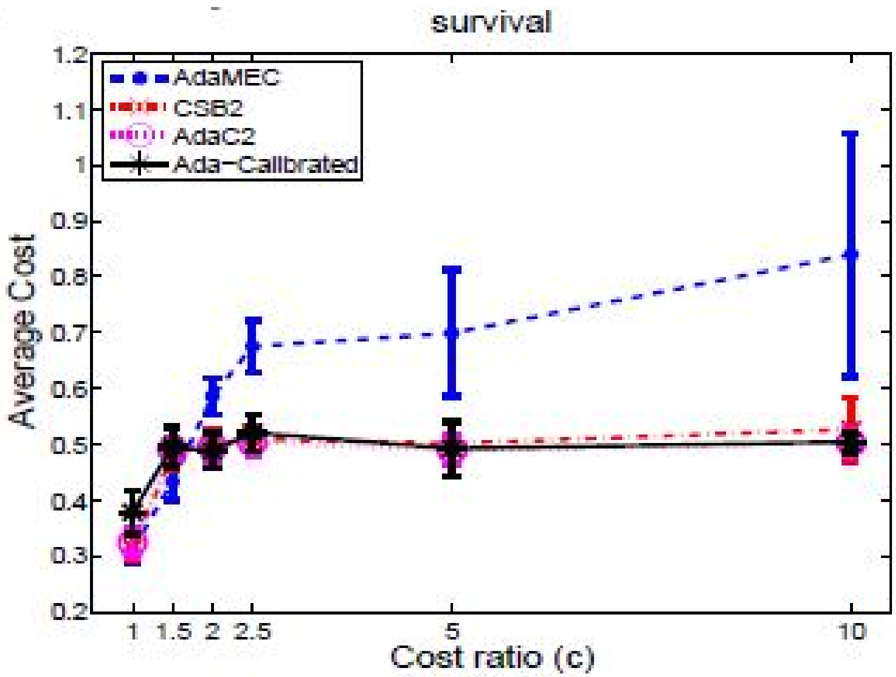
- Weak learner: univariate logistic regression

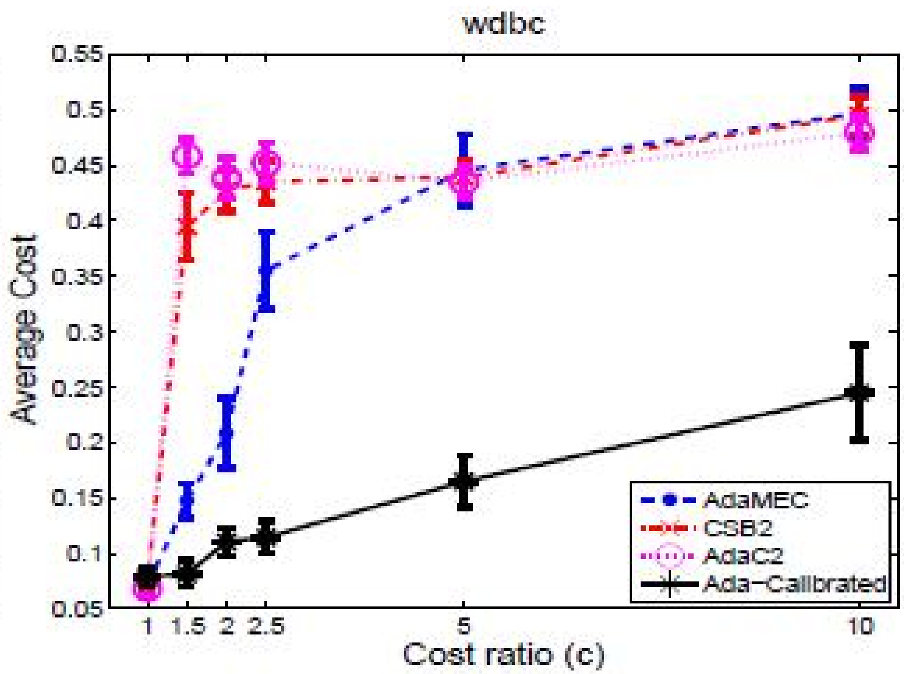
- 7 datasets

- Evaluation: average cost, precision, recall, f1-measure

- Skew ratios: $c = \{1, 1.5, 2, 2.5, 5, 10\}$

Empirical Results (1)

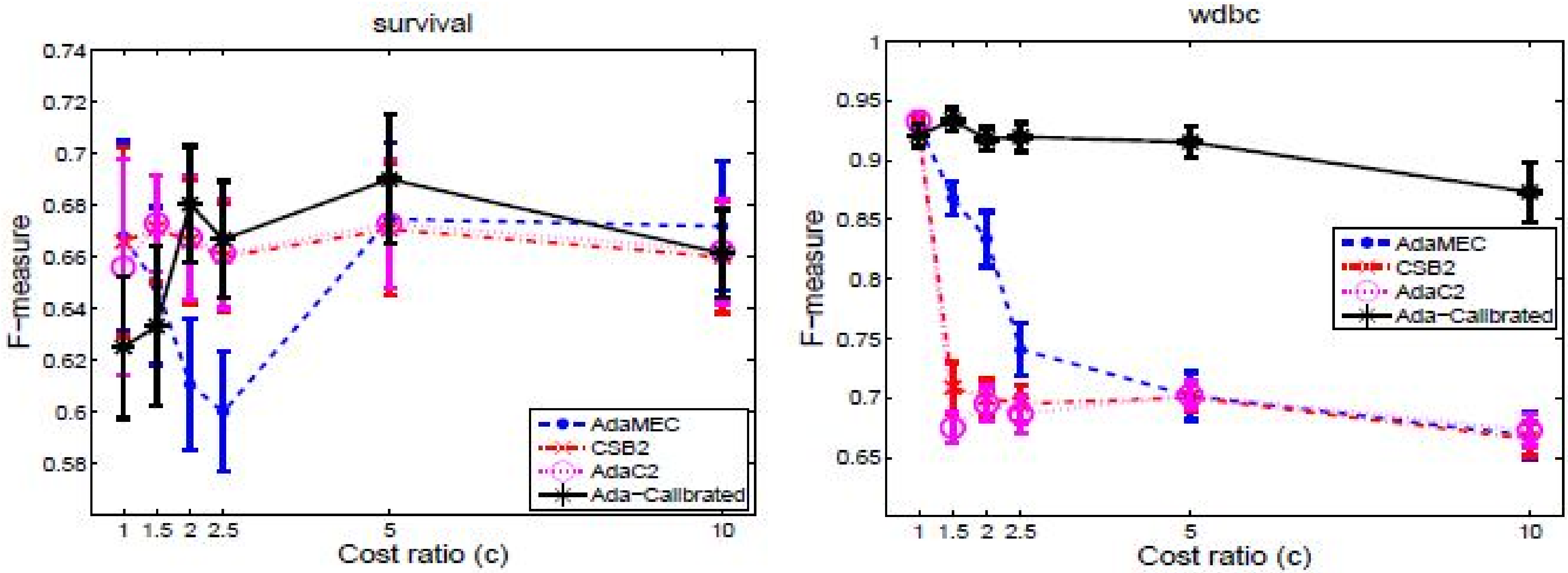




All methods equivalent when $c = 1$ (no skew)

Smaller datasets: **Ada-Calibrated** comparable to rest
Larger datasets: **Ada-Calibrated** superior to rest

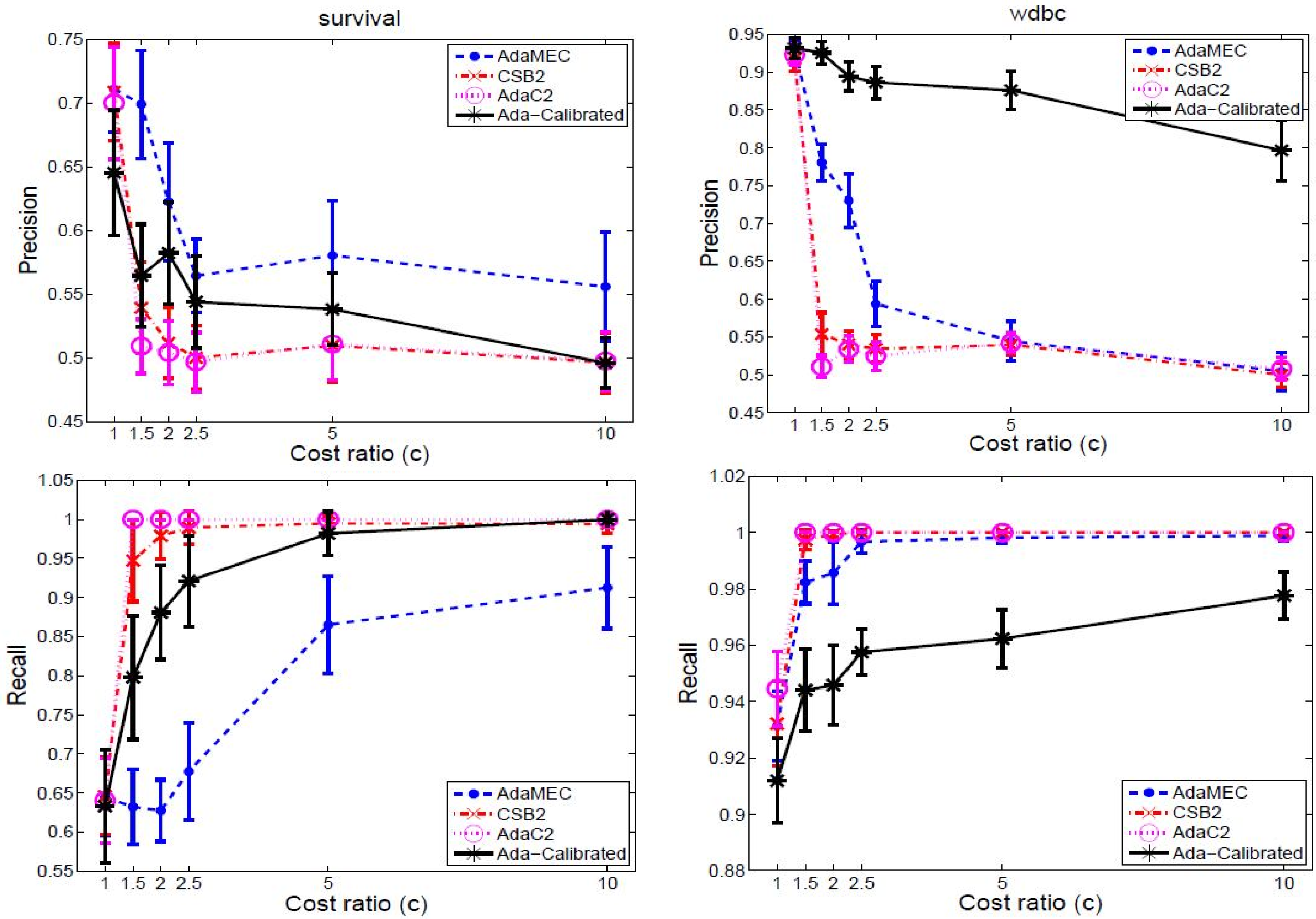
Empirical Results (2)



All methods equivalent when $c = 1$ (no skew)

Smaller datasets: **Ada-Calibrated** comparable to rest
Larger datasets: **Ada-Calibrated** superior to rest

Empirical Results (3)



Conclusion

- Calibrating AdaBoost empirically **comparable** (small data) or **superior** (big data) to alternatives published 1998 - 2011
- Conceptual **simplicity**; no need for new algorithms, or hyperparameter setting
- **No need to retrain** if skew ratio changes
- Retains **theoretical guarantees** of AdaBoost & decision theory

Thank you!
Danke!