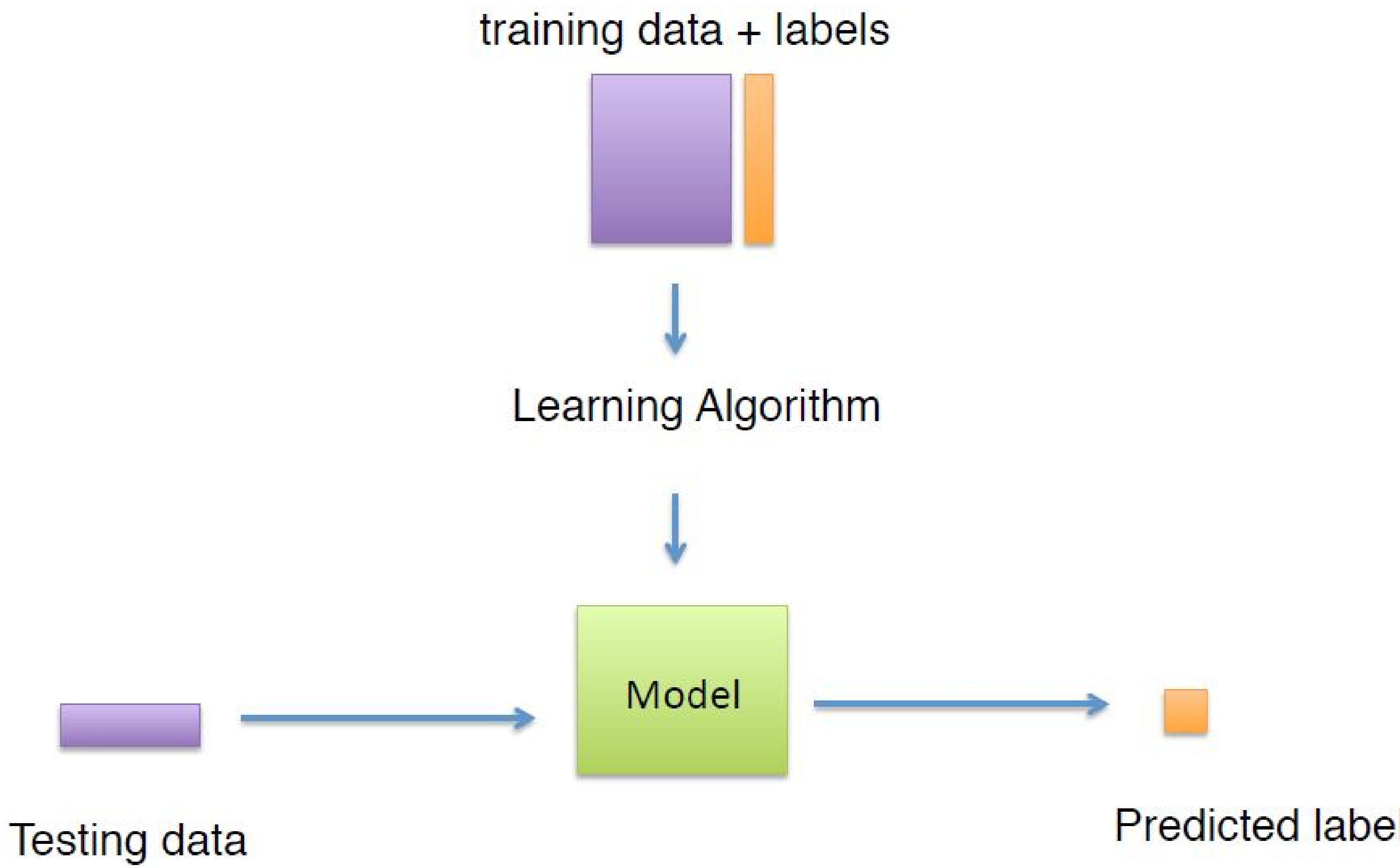
Cost-sensitive boosting algorithms: do we really need them?

Nikos Nikolaou

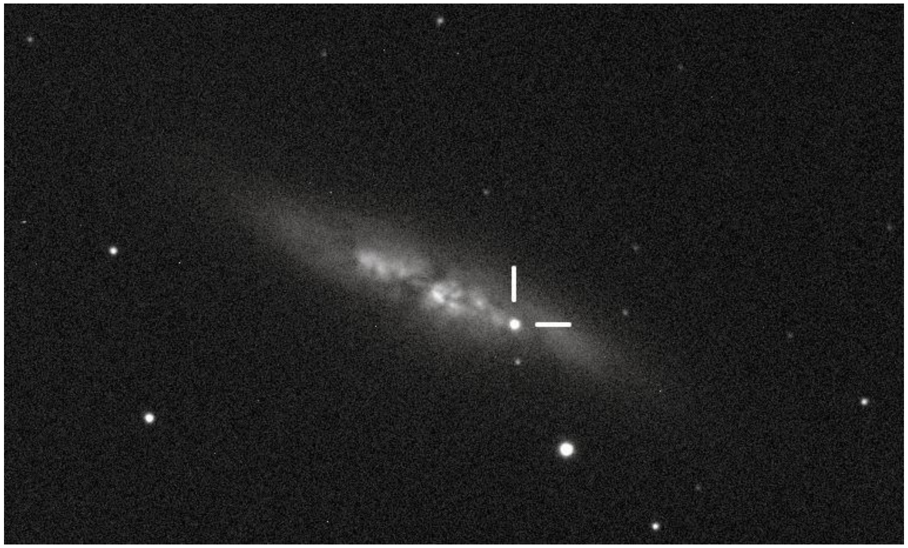
The University of Manchester

Supervised learning



Asymmetric learning

Cost-sensitive
different errors have
have different costs

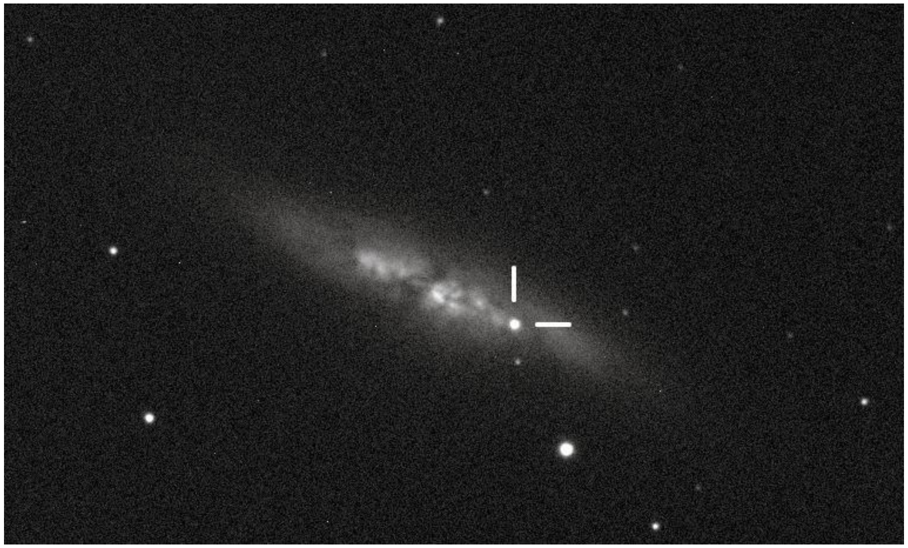


Imbalanced classes
different classes appear
with different frequency

...or both!

Asymmetric learning

Cost-sensitive
different errors have
have different costs



Imbalanced classes
different classes appear
with different frequency

...or both!

Boosting \ AdaBoost

- Ensemble technique: sequentially combine multiple weak learners to build a strong one

Boosting \ AdaBoost

- Ensemble technique: sequentially combine multiple weak learners to build a strong one

Top 10 algorithms in data mining

Xindong Wu · Vipin Kumar · J. Ross Quinlan · Joydeep Ghosh · Qiang Yang · Hiroshi Motoda · Geoffrey J. McLachlan · Angus Ng · Bing Liu · Philip S. Yu · Zhi-Hua Zhou · Michael Steinbach · David J. Hand · Dan Steinberg

Received: 9 July 2007 / Revised: 28 September 2007 / Accepted: 8 October 2007
Published online: 4 December 2007

Abstract This paper presents the top 10 data mining algorithms identified by the IEEE International Conference on Data Mining (ICDM) in December 2006: C4.5, *k*-Means, SVM, Apriori, EM, PageRank, AdaBoost, *k*NN, Naive Bayes, and CART. These top 10 algorithms are among the most influential data mining algorithms in the research community. With each algorithm, we provide a description of the algorithm, discuss the impact of the algorithm, and review current and further research on the algorithm. These 10 algorithms cover classification,

Boosting \ AdaBoost

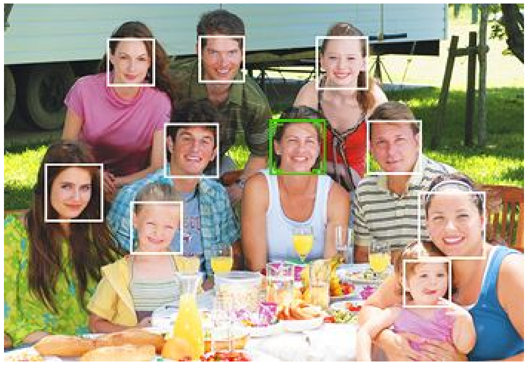
- Ensemble technique: sequentially combine multiple weak learners to build a strong one

Top 10 algorithms in data mining

Xindong Wu · Vipin Kumar · J. Ross Quinlan · Joydeep Ghosh · Qiang Yang · Hiroshi Motoda · Geoffrey J. McLachlan · Angus Ng · Bing Liu · Philip S. Yu · Zhi-Hua Zhou · Michael Steinbach · David J. Hand · Dan Steinberg

Received: 9 July 2007 / Revised: 28 September 2007 / Accepted: 8 October 2007
Published online: 4 December 2007

Abstract This paper presents the top 10 data mining algorithms identified by the IEEE International Conference on Data Mining (ICDM) in December 2006: C4.5, *k*-Means, SVM, Apriori, EM, PageRank, AdaBoost, *k*NN, Naive Bayes, and CART. These top 10 algorithms are among the most influential data mining algorithms in the research community. With each algorithm, we provide a description of the algorithm, discuss the impact of the algorithm, and review current and further research on the algorithm. These 10 algorithms cover classification,



Face recognition in phone cameras

Boosting \ AdaBoost

- Ensemble technique: sequentially combine multiple weak learners to build a strong one

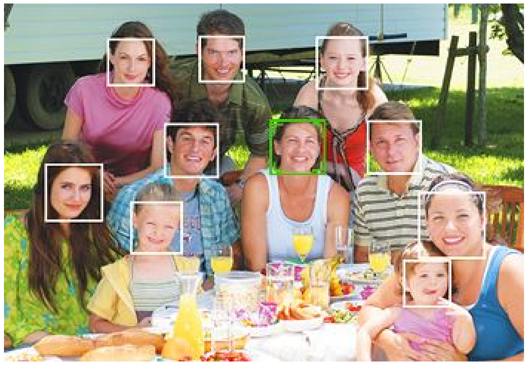
Top 10 algorithms in data mining

Xindong Wu · Vipin Kumar · J. Ross Quinlan · Joydeep Ghosh · Qiang Yang · Hiroshi Motoda · Geoffrey J. McLachlan · Angus Ng · Bing Liu · Philip S. Yu · Zhi-Hua Zhou · Michael Steinbach · David J. Hand · Dan Steinberg

Received: 9 July 2007 / Revised: 28 September 2007 / Accepted: 8 October 2007
Published online: 4 December 2007

Abstract This paper presents the top 10 data mining algorithms identified by the IEEE International Conference on Data Mining (ICDM) in December 2006: C4.5, *k*-Means, SVM, Apriori, EM, PageRank, AdaBoost, *k*NN, Naive Bayes, and CART. These top 10 algorithms are among the most influential data mining algorithms in the research community. With each algorithm, we provide a description of the algorithm, discuss the impact of the algorithm, and review current and further research on the algorithm. These 10 algorithms cover classification,

kaggle



Face recognition in phone cameras

Boosting \ AdaBoost

- Ensemble technique: sequentially combine multiple weak learners to build a strong one

An Empirical Comparison of Supervised Learning Algorithms

Rich Caruana CARUANA@CS.CORNELL.EDU
Alexandru Niculescu-Mizil ALEXN@CS.CORNELL.EDU
Department of Computer Science, Cornell University, Ithaca, NY 14853 USA

With excellent performance on all eight metrics, calibrated boosted trees were the best learning algorithm overall. Random forests are close second, followed by uncalibrated bagged trees, calibrated SVMs, and uncalibrated neural nets. The models that performed

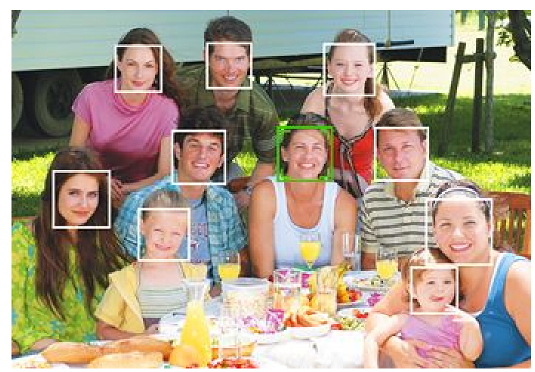
Top 10 algorithms in data mining

Xindong Wu · Vipin Kumar · J. Ross Quinlan · Joydeep Ghosh · Qiang Yang · Hiroshi Motoda · Geoffrey J. McLachlan · Angus Ng · Bing Liu · Philip S. Yu · Zhi-Hua Zhou · Michael Steinbach · David J. Hand · Dan Steinberg

Received: 9 July 2007 / Revised: 28 September 2007 / Accepted: 8 October 2007
Published online: 4 December 2007

Abstract This paper presents the top 10 data mining algorithms identified by the IEEE International Conference on Data Mining (ICDM) in December 2006: C4.5, *k*-Means, SVM, Apriori, EM, PageRank, AdaBoost, *k*NN, Naive Bayes, and CART. These top 10 algorithms are among the most influential data mining algorithms in the research community. With each algorithm, we provide a description of the algorithm, discuss the impact of the algorithm, and review current and further research on the algorithm. These 10 algorithms cover classification,

kaggle



Face recognition in phone cameras

Boosting \ AdaBoost

- Ensemble technique: sequentially combine multiple weak learners to build a strong one

An Empirical Comparison of Supervised Learning Algorithms

Rich Caruana CARUANA@CS.CORNELL.EDU
Alexandru Niculescu-Mizil ALEXN@CS.CORNELL.EDU
Department of Computer Science, Cornell University, Ithaca, NY 14853 USA

With excellent performance on all eight metrics, calibrated boosted trees were the best learning algorithm overall. Random forests are close second, followed by uncalibrated bagged trees, calibrated SVMs, and uncalibrated neural nets. The models that performed

Top 10 algorithms in data mining

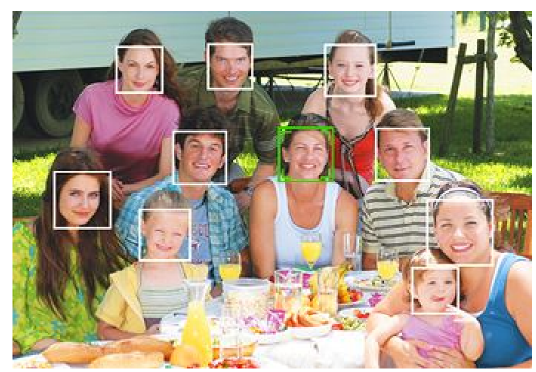
Xindong Wu · Vipin Kumar · J. Ross Quinlan · Joydeep Ghosh · Qiang Yang · Hiroshi Motoda · Geoffrey J. McLachlan · Angus Ng · Bing Liu · Philip S. Yu · Zhi-Hua Zhou · Michael Steinbach · David J. Hand · Dan Steinberg

Received: 9 July 2007 / Revised: 28 September 2007 / Accepted: 8 October 2007
Published online: 4 December 2007

Abstract This paper presents the top 10 data mining algorithms identified by the IEEE International Conference on Data Mining (ICDM) in December 2006: C4.5, *k*-Means, SVM, Apriori, EM, PageRank, AdaBoost, *k*NN, Naive Bayes, and CART. These top 10 algorithms are among the most influential data mining algorithms in the research community. With each algorithm, we provide a description of the algorithm, discuss the impact of the algorithm, and review current and further research on the algorithm. These 10 algorithms cover classification,

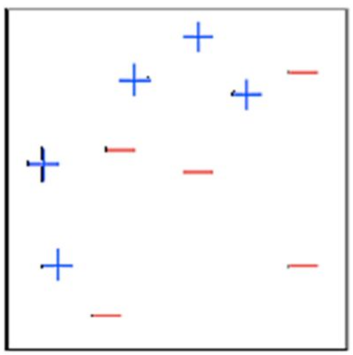
2003 Gödel Prize

kaggle

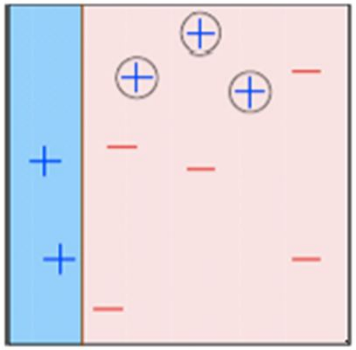


Face recognition in phone cameras

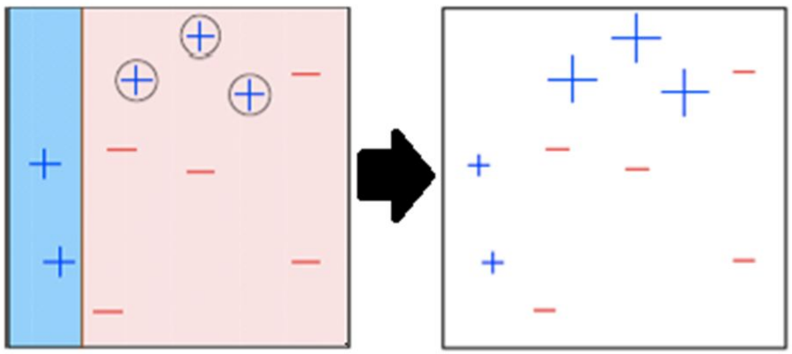
AdaBoost example



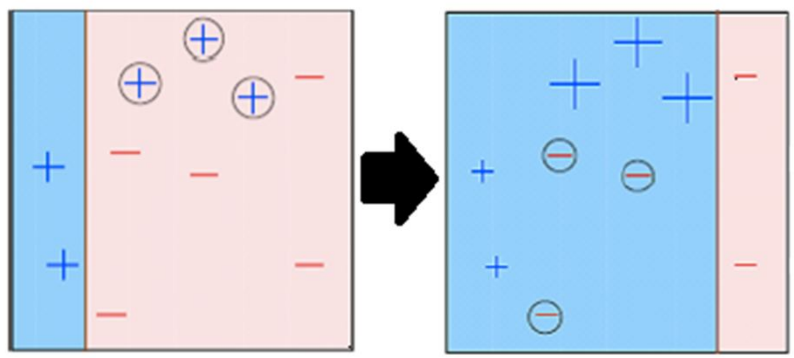
AdaBoost example



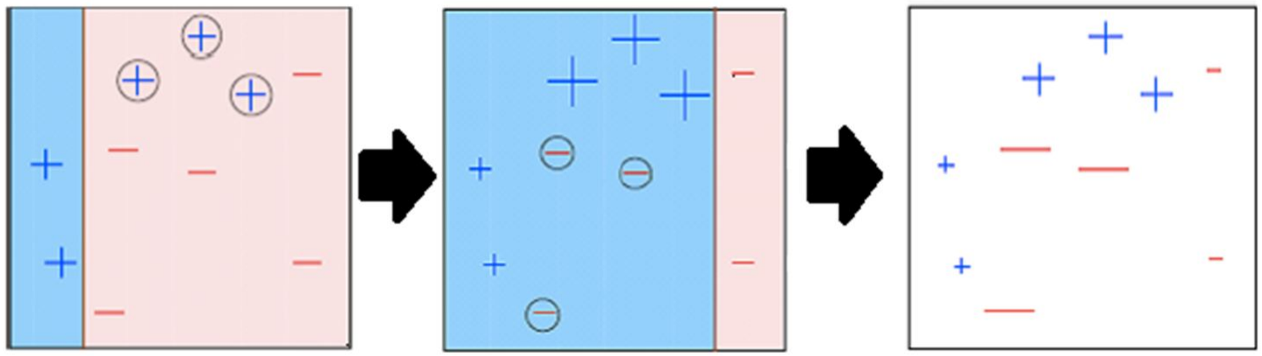
AdaBoost example



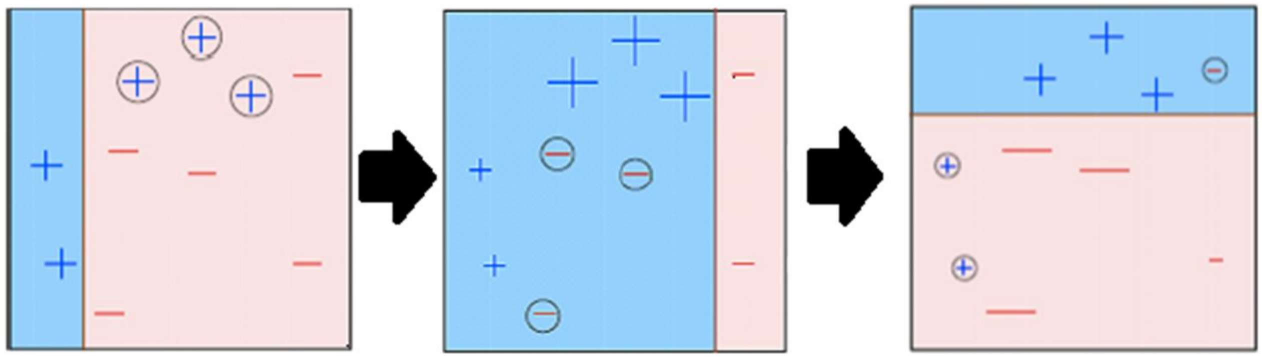
AdaBoost example



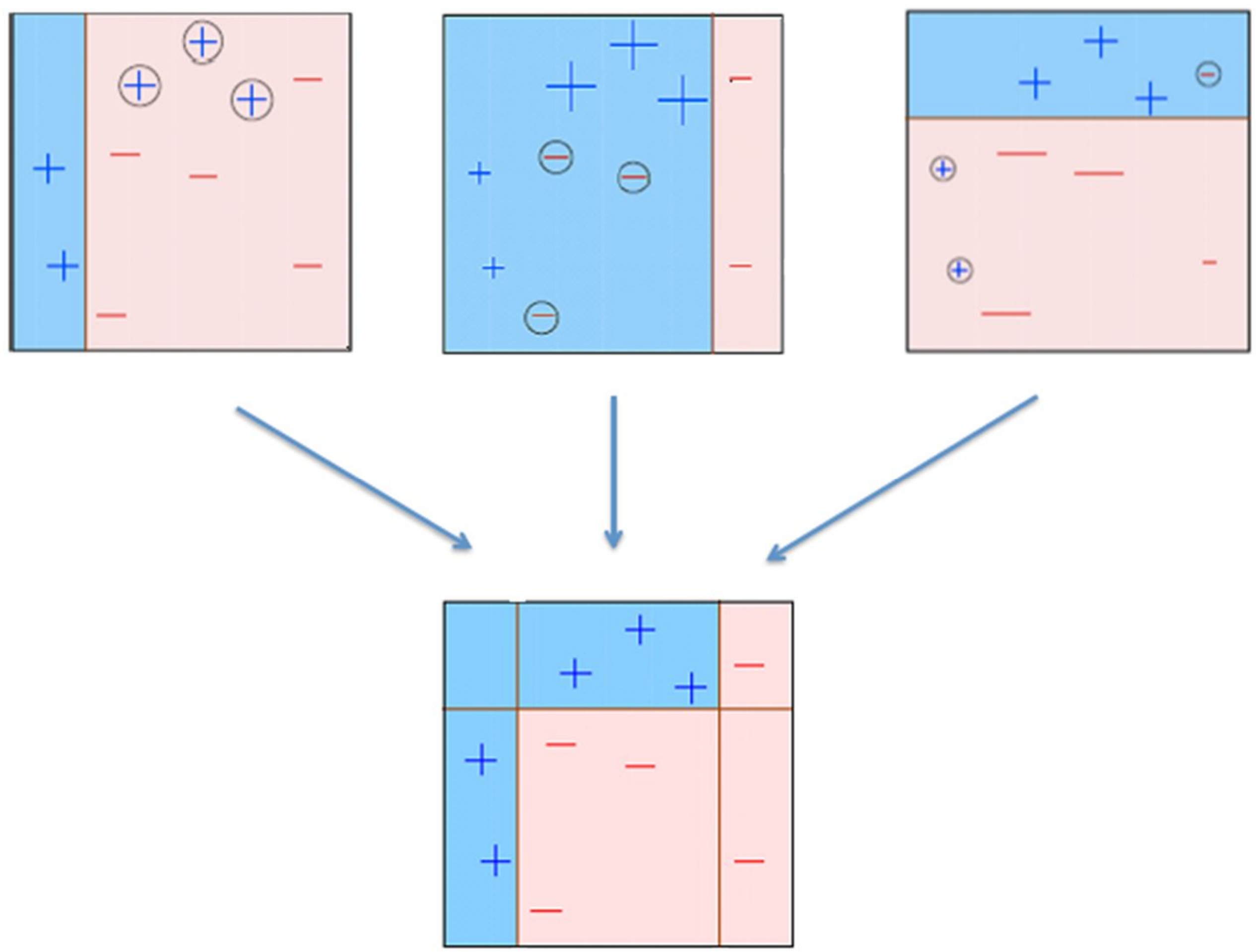
AdaBoost example



AdaBoost example



AdaBoost example



AdaBoost under the hood

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

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

Assign a confidence score to each weak learner

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

Update examples' weights

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight distribution over the examples

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

Confidence weighted majority vote

AdaBoost under the hood

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

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

Assign a confidence score to each weak learner

Can it handle cost-sensitive problems?

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

Update examples' weights

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight distribution over the examples

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

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

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

Update examples' weights

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight
distribution over the examples

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

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

(Ting & Zheng, 1998)

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

Update examples' weights

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight distribution over the examples

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

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)

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

Update examples' weights

(Ting & Zheng, 1998)

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight distribution over the examples

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

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)

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

Update examples' weights

(Ting & Zheng, 1998)

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight distribution over the examples

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

Confidence weighted majority vote

Asymmetric boosting variants

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

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

(Fan et al., 1999)
(Cohen & Singer, 1999)
(Ting, 2000)

Assign a confidence score to each weak learner

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

(Ting & Zheng, 1998)
(Ting, 2000)

Update examples' weights

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight distribution over the examples

(Ting, 2000)

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

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)

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

Update examples' weights

(Ting & Zheng, 1998)
(Ting, 2000)

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight distribution over the examples

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

Confidence weighted majority vote

(Ting, 2000)

Asymmetric boosting variants

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

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

Assign a confidence score to each weak learner

(Fan et al., 1999)
(Cohen & Singer, 1999)
(Ting, 2000)
(Joshi et al., 2001)

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

Update examples' weights

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

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight distribution over the examples

(Ting, 2000)

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

Confidence weighted majority vote

Asymmetric boosting variants

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

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

(Fan et al., 1999)
(Cohen & Singer, 1999)
(Ting, 2000)
(Joshi et al., 2001)
(Sun et al., 2005; 2007)

Assign a confidence score to each weak learner

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

Update examples' weights

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

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight distribution over the examples

(Ting, 2000)

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

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)

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight distribution over the examples

(Ting, 2000)

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

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)

$$D_i^1 = \frac{1}{N}$$

Start with a uniform weight distribution over the examples

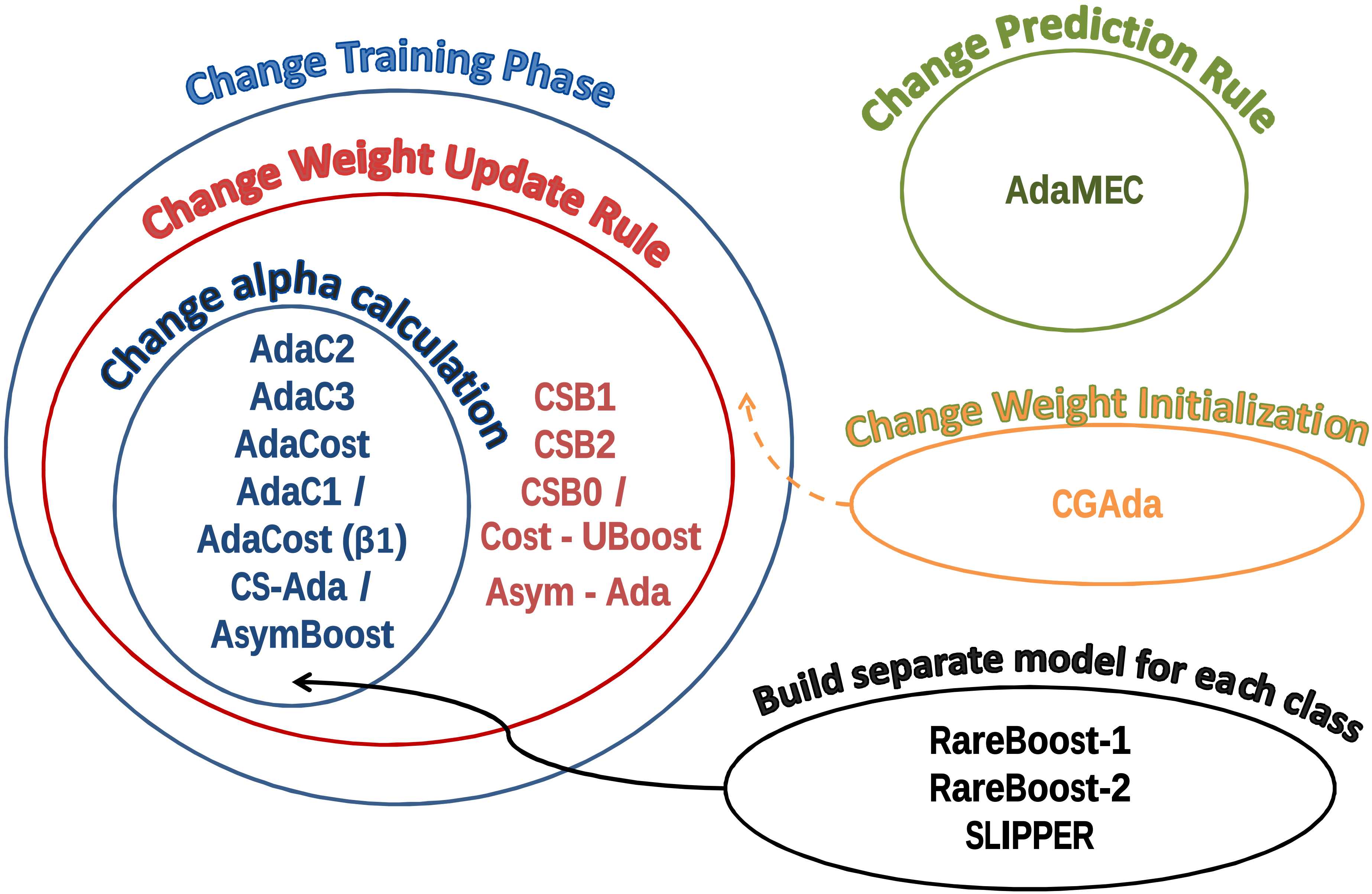
(Landesa-Vázquez & Alba-Castro, 2013;2015a;2015b)

(Ting, 2000)

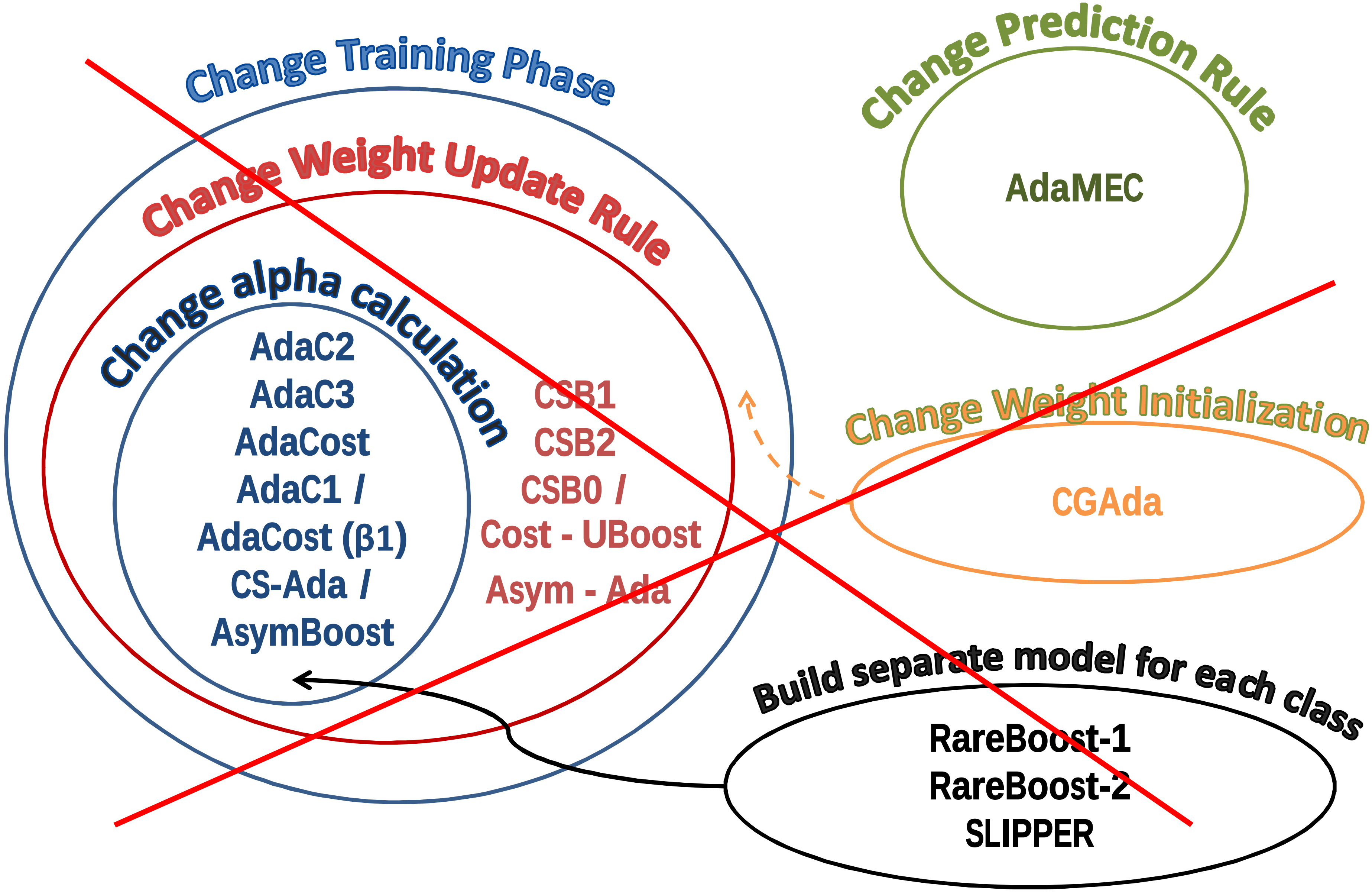
$$H(\mathbf{x}') = sign\left[\sum_{t=1}^{M}\alpha_t h_t(\mathbf{x}')\right]$$

Confidence weighted majority vote

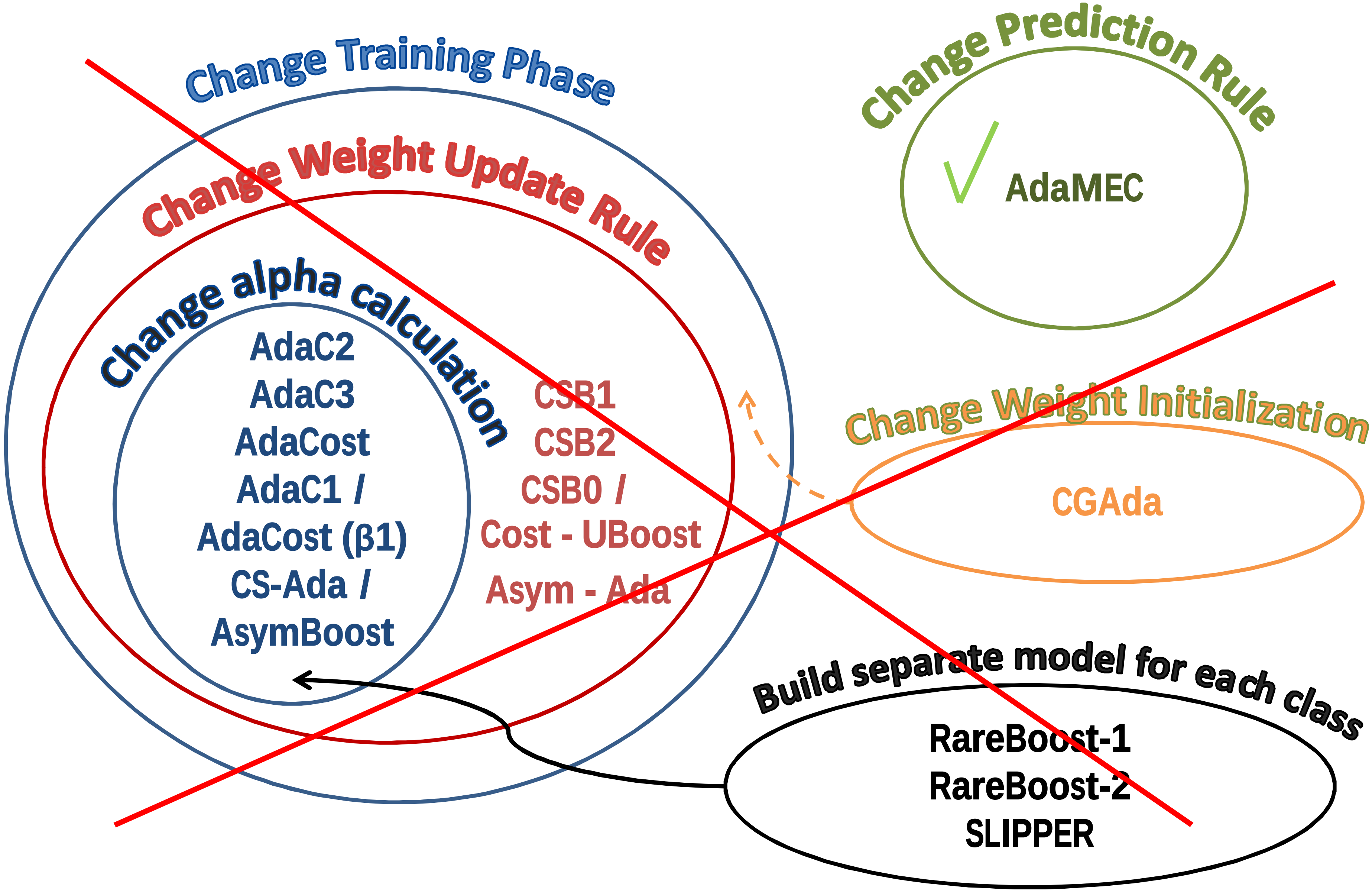
Asymmetric boosting variants



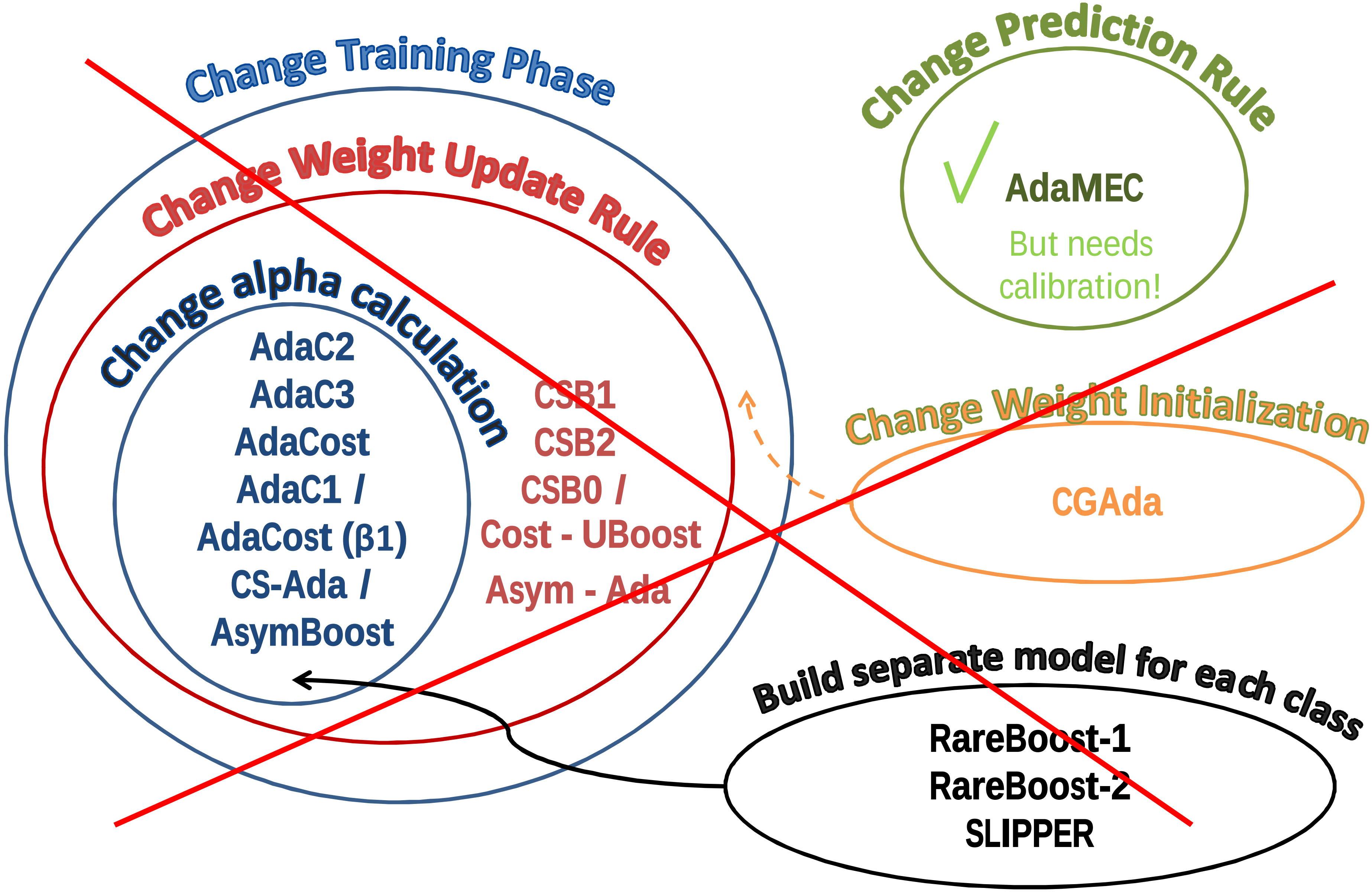
Asymmetric boosting variants



Asymmetric boosting variants



Asymmetric boosting variants



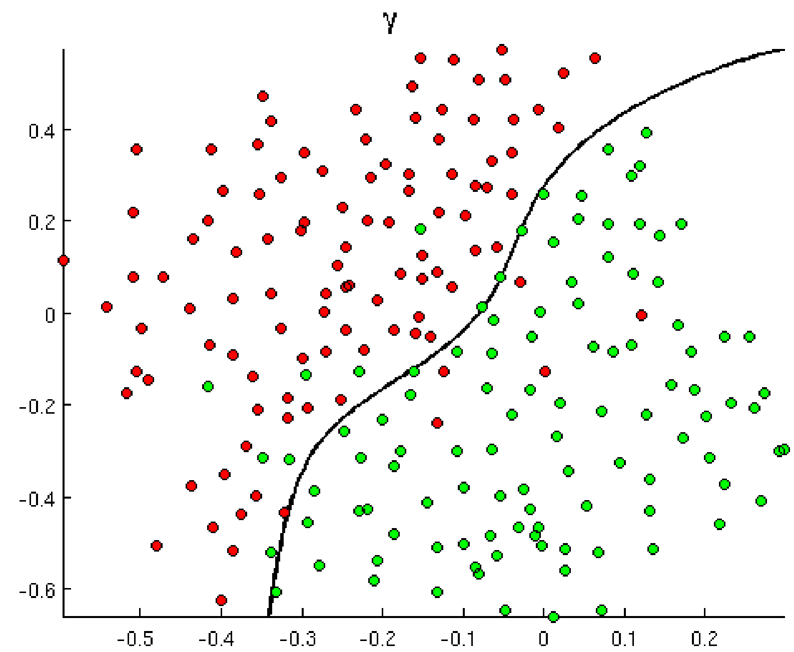
Issues with modifying training phase

- Lack **theoretical guarantees** of original AdaBoost
- Most **heuristic**, no decision-theoretic motivation
- Need to **retrain** if skew ratio changes
- Require **extra hyperparameters** to be set via CV

Issues with *AdaMEC*

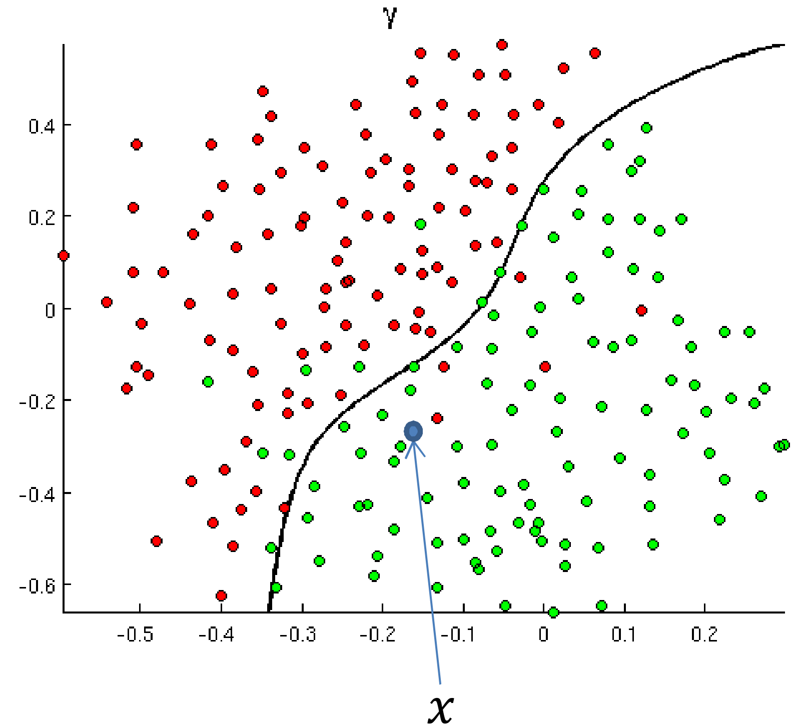
- Changes prediction rule to **minimum expected cost** √
- Problem: **incorrectly assumes scores** are **probability estimates**...
- ...but can correct this via **calibration**

Things classifiers do...



Things classifiers do…

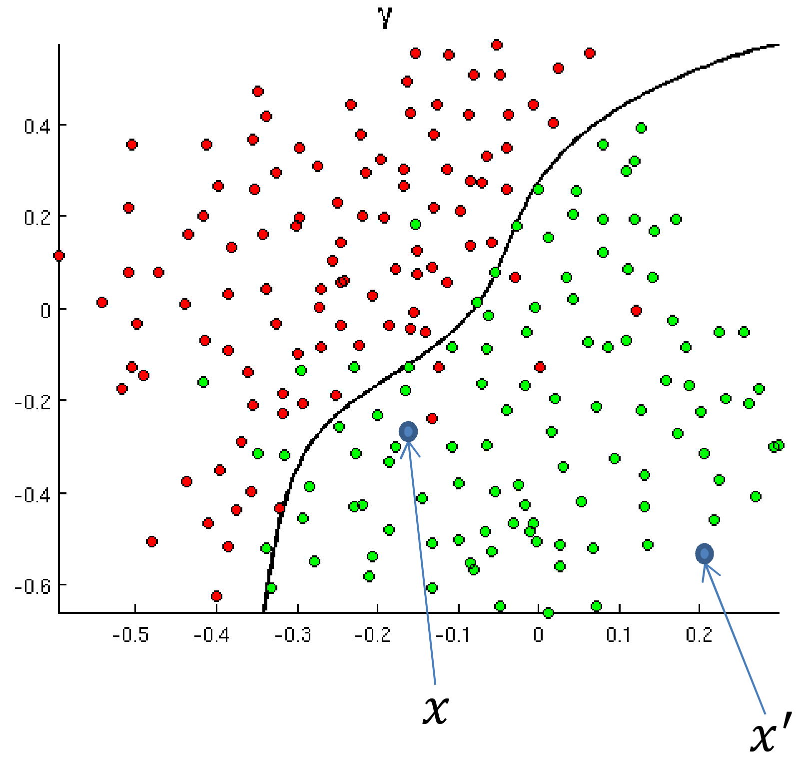
- **Classify** examples
  - Is x positive?



Things classifiers do...

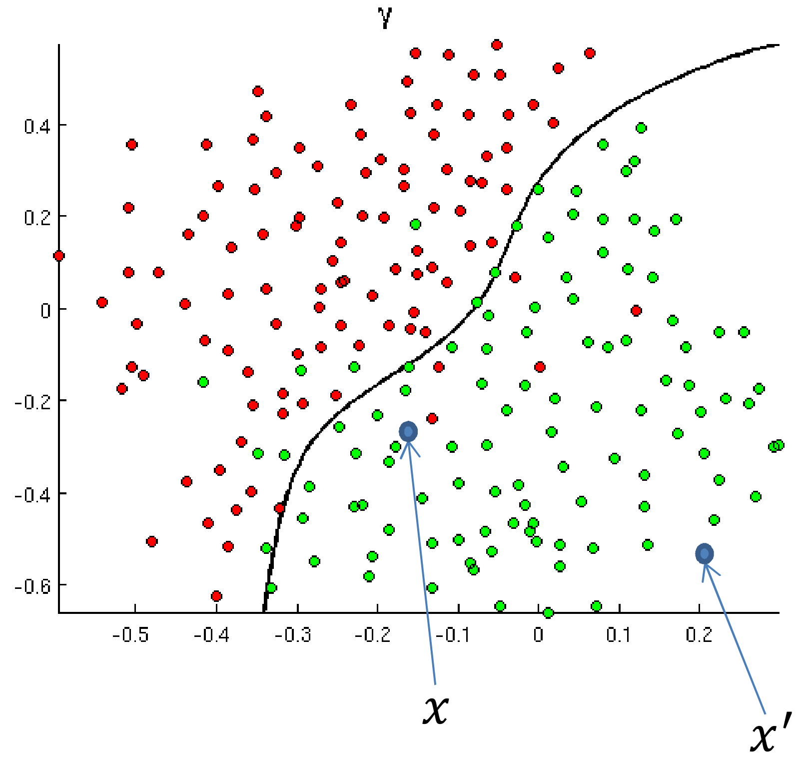
- **Classify** examples
  - Is x positive?

- **Rank** examples
  - Is x 'more positive' than x'?



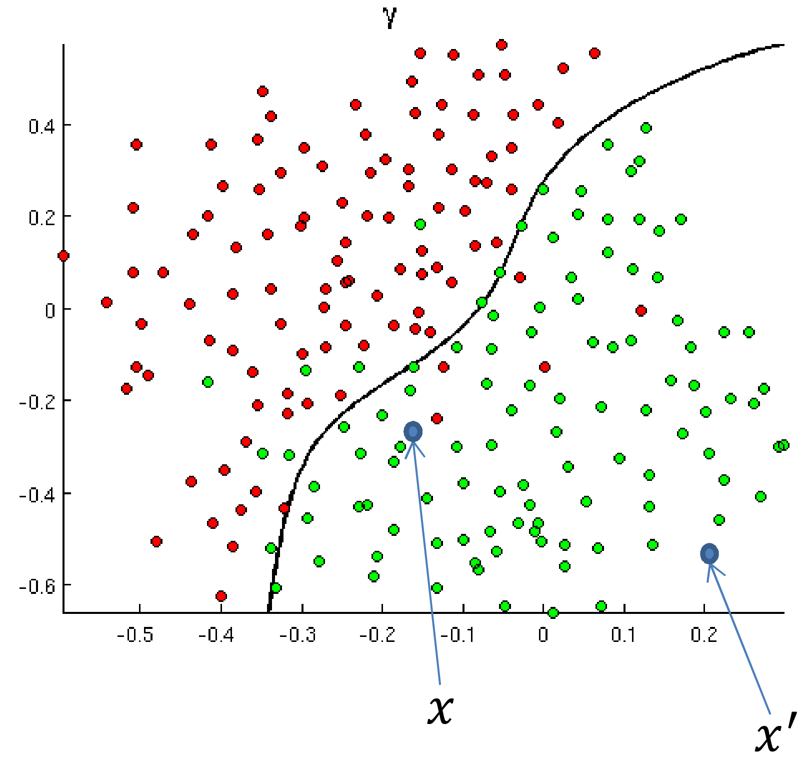
Things classifiers do...

- **Classify** examples
  - Is x positive?
- **Rank** examples
  - Is x 'more positive' than x'?
- Output a **score** for each example
  - 'How positive' is x?

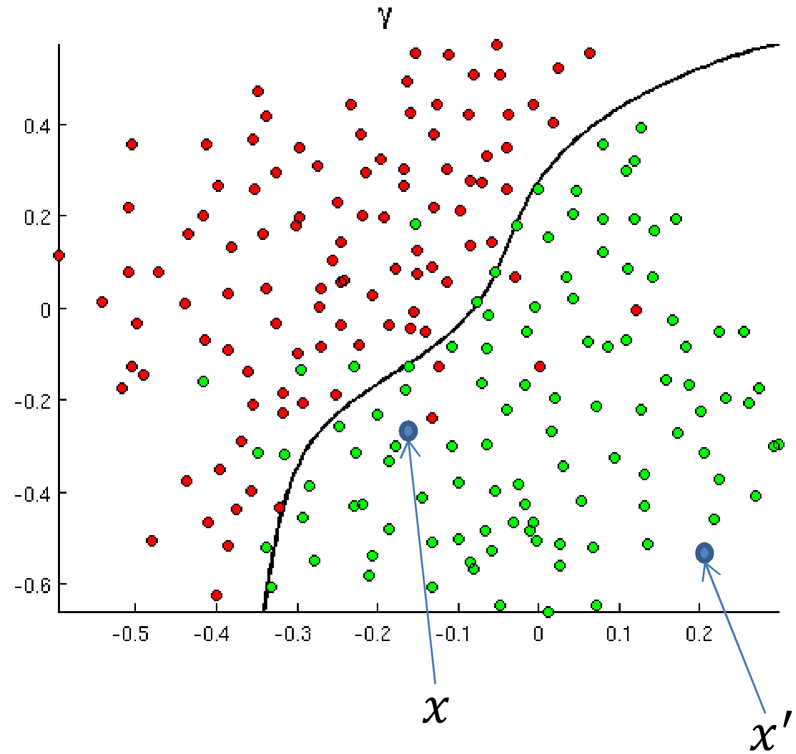


Things classifiers do…

- **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?



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?

calibration

- 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 x to class that minimizes **expected** cost, i.e. to Pos only if:

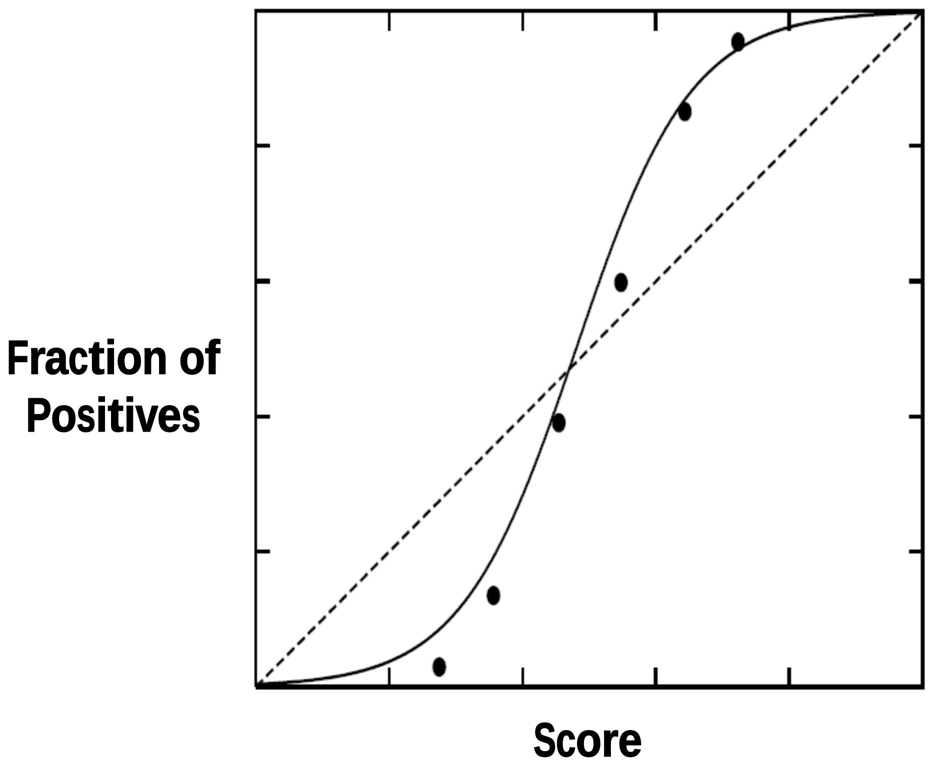
expected cost of assigning x to Pos < expected cost of assigning x to Neg

$$\Leftrightarrow$$

$$\hat{p}(y = 1|x) > \underbrace{\frac{C_{FP}}{C_{FN} + C_{FP}}}_{\text{Costs are part of problem definition}}$$

Probability estimates of AdaBoost

Score for Boosting: $s(\mathbf{x}) = \frac{\sum_{t:h_t(x)=1} \alpha_t}{\sum_{t=1}^{M} \alpha_t}$ $\in [0, 1]$

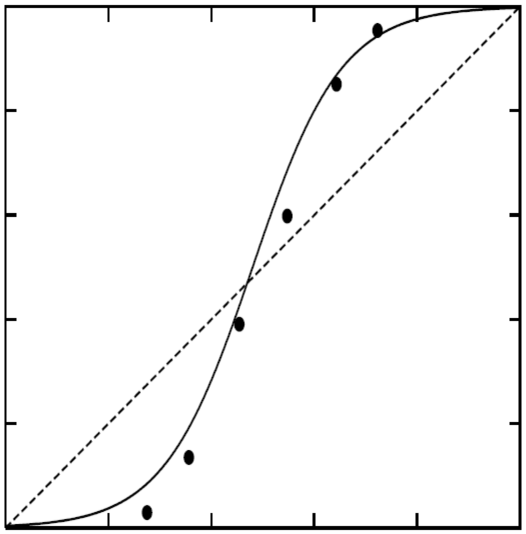


Boosted trees / stumps: **sigmoid distortion**; scores pushed more towards 0 or 1 as num. of boosting rounds increases

(Niculescu-Mizil & Caruana, 2006)

Calibrating AdaBoost: 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
- Alternative: isotonic regression

Calibrating AdaBoost for cost-sensitive learning

On training set:

- Train AdaBoost ensemble H_M

On validation set:

- Calculate score $s(\mathbf{x}) = \frac{\sum_{t:h_t(x)=1} \alpha_t}{\sum_{t=1}^{M} \alpha_t} \in [0,1]$ 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 $H_M(\mathbf{x}) = sign\ [\hat{p}(y = 1|x) - \frac{C_{FP}}{C_{FP}+C_{FN}}]$

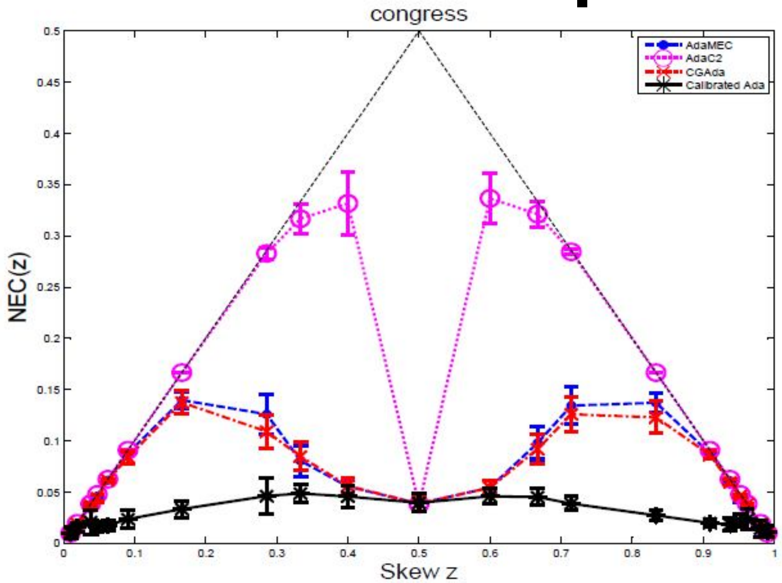
Experimental design

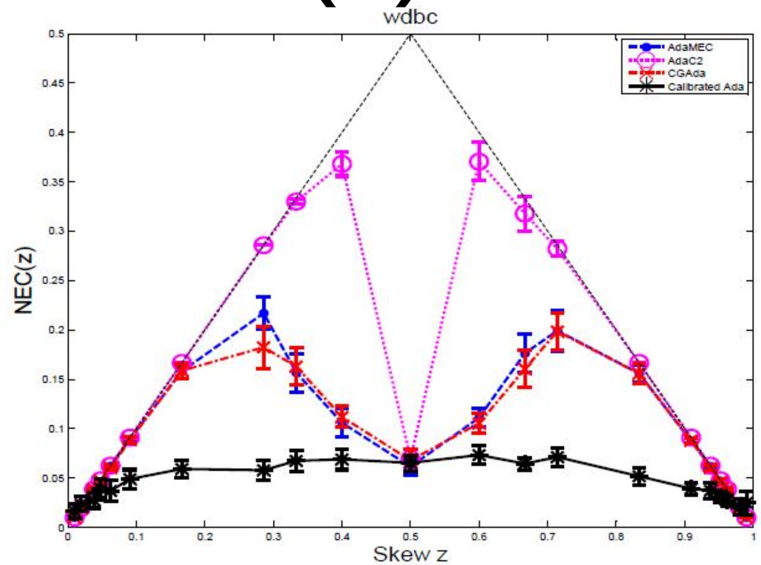
- AdaC2 vs. CGAda vs. AdaMEC vs. Calibrated AdaBoost

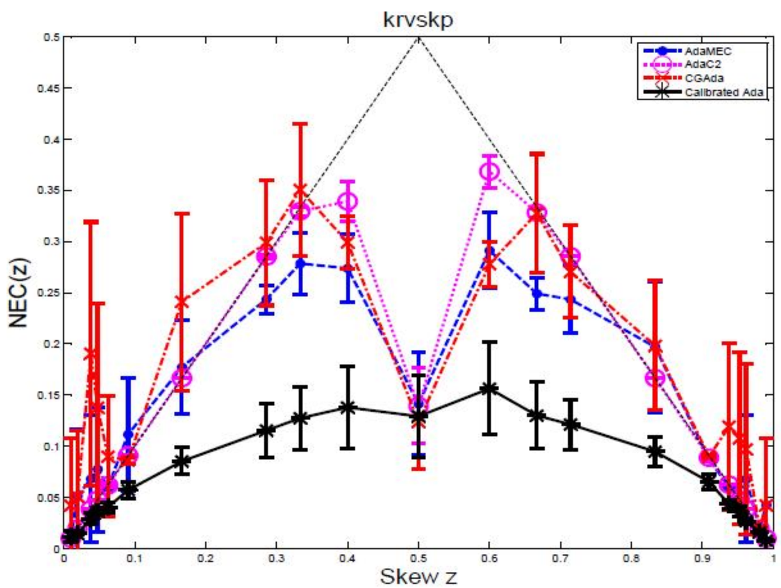
  75% Tr / 25% Te 50% Tr / 25% Cal / 25% Te

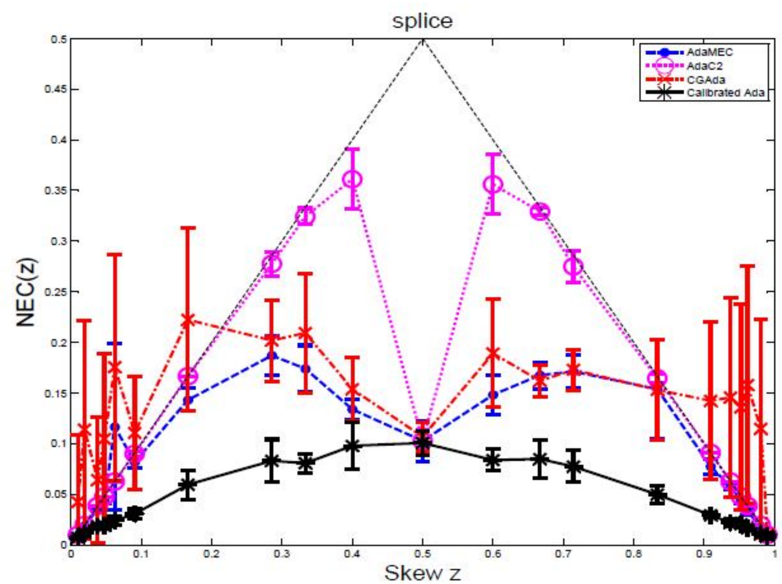
- Weak learner: univariate logistic regression
- 18 datasets
- Evaluation: normalized expected cost $\in [0, 1]$
- Various skew ratios: $z = \frac{c_{FP}}{c_{FN} + c_{FP}}$

Empirical results (1)



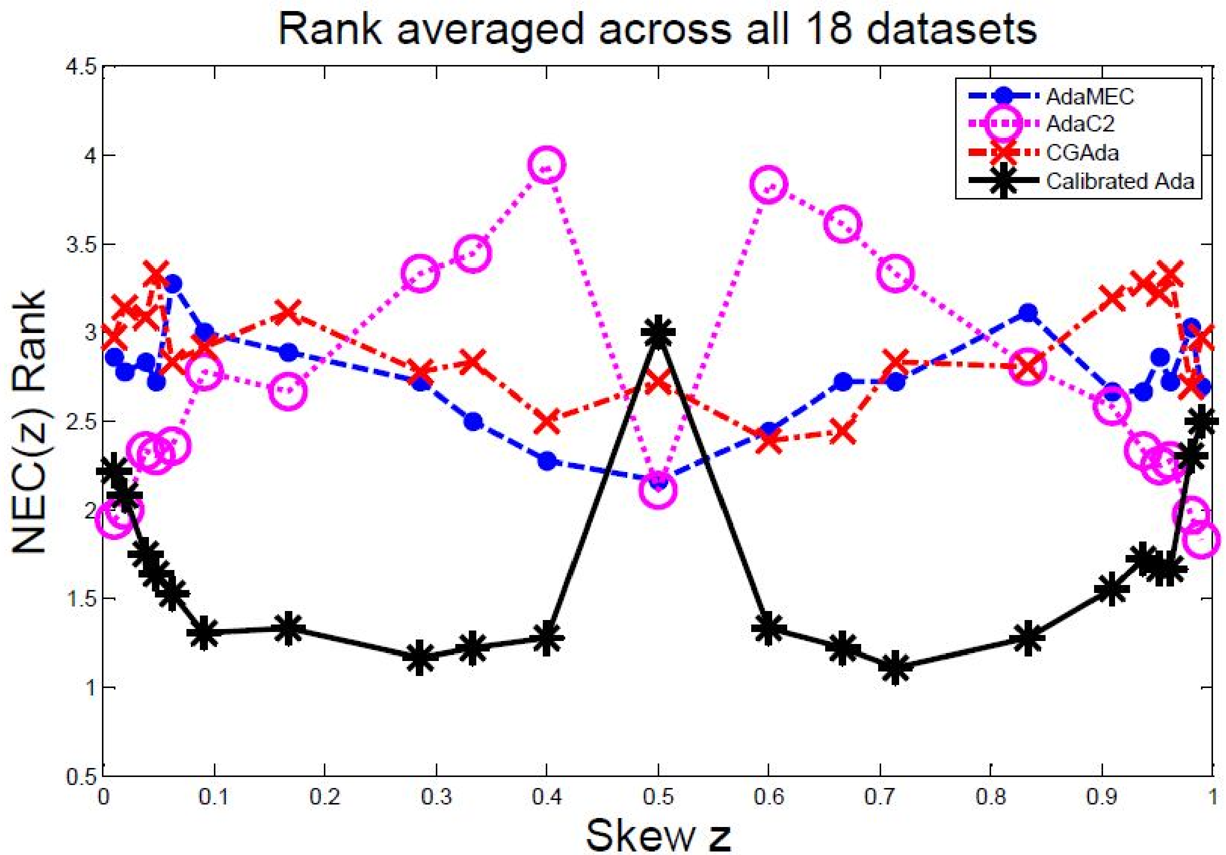




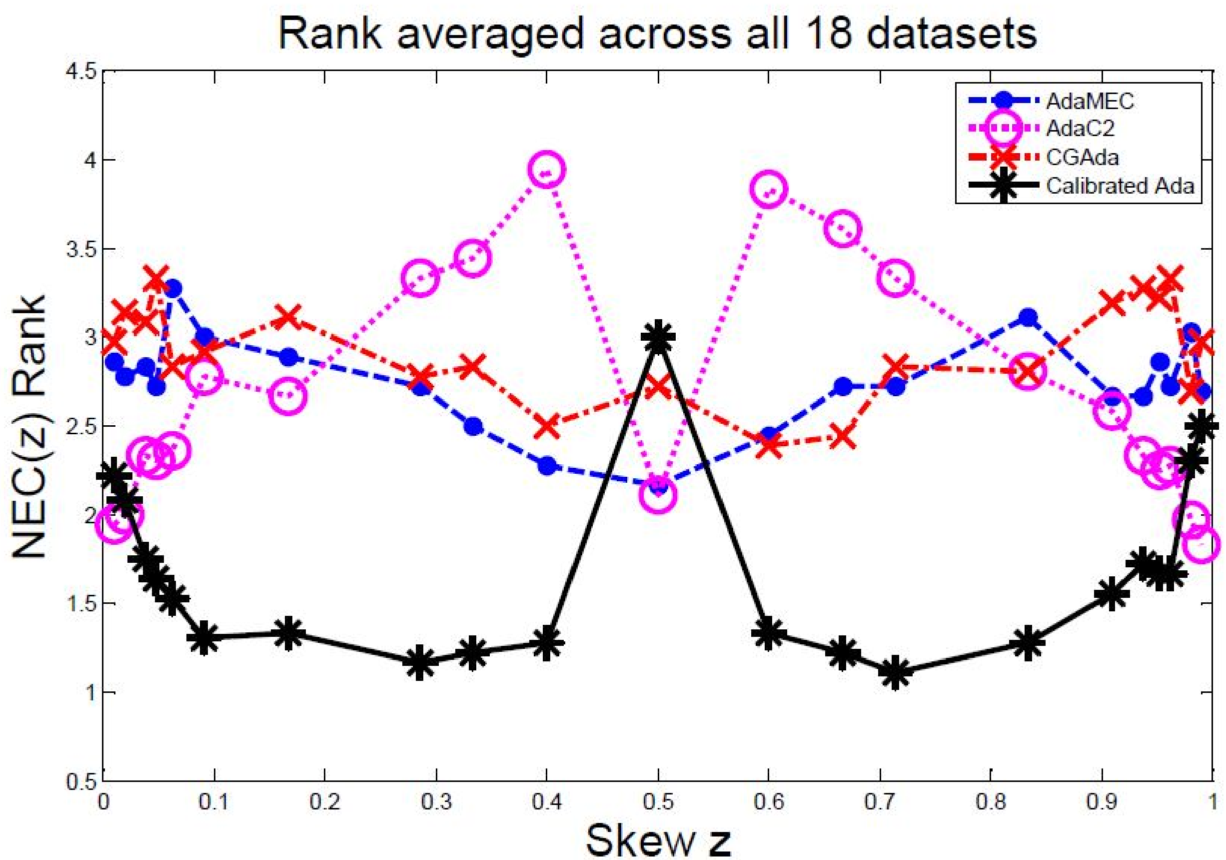


Ada-Calibrated at least as good as best, especially good on larger datasets

Empirical results (2)

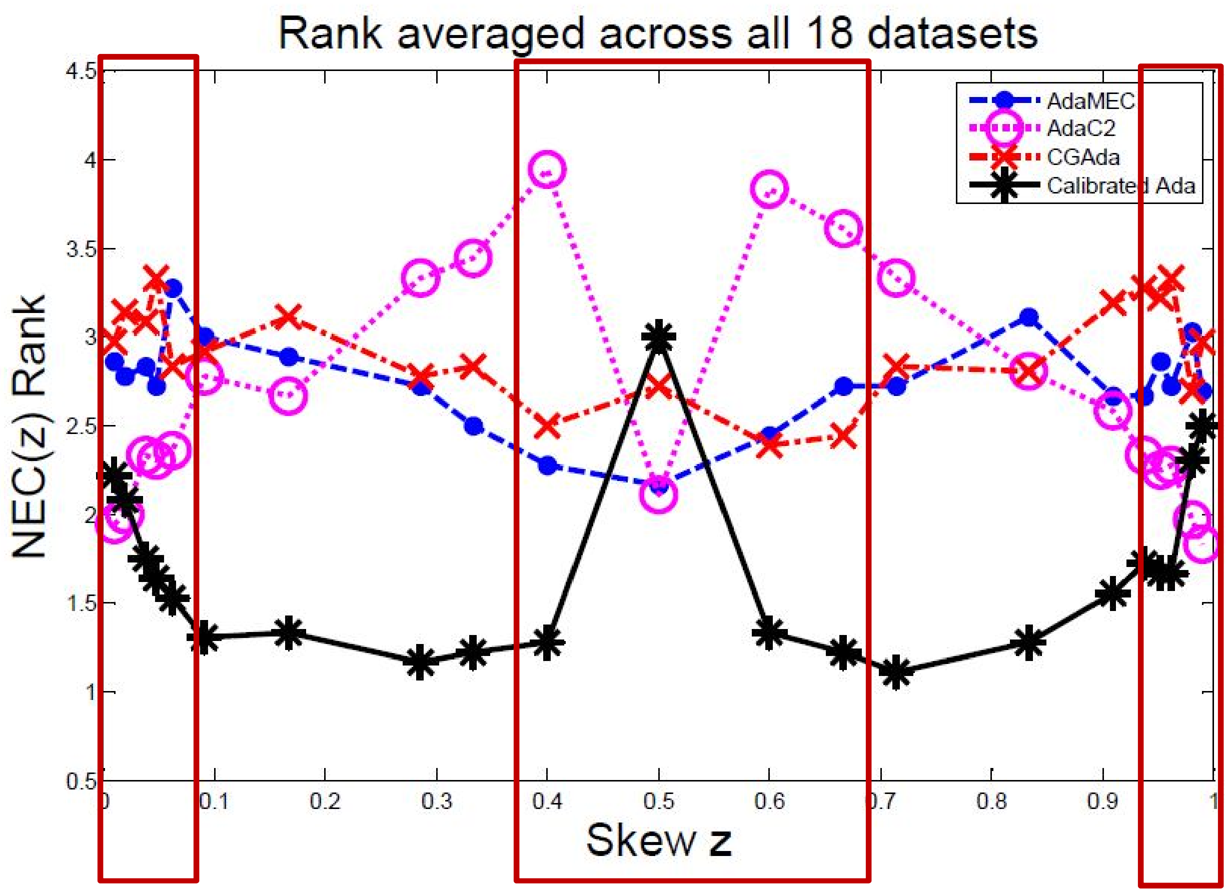


Empirical results (2)



Nemenyi test at the 0.05 level on the differences

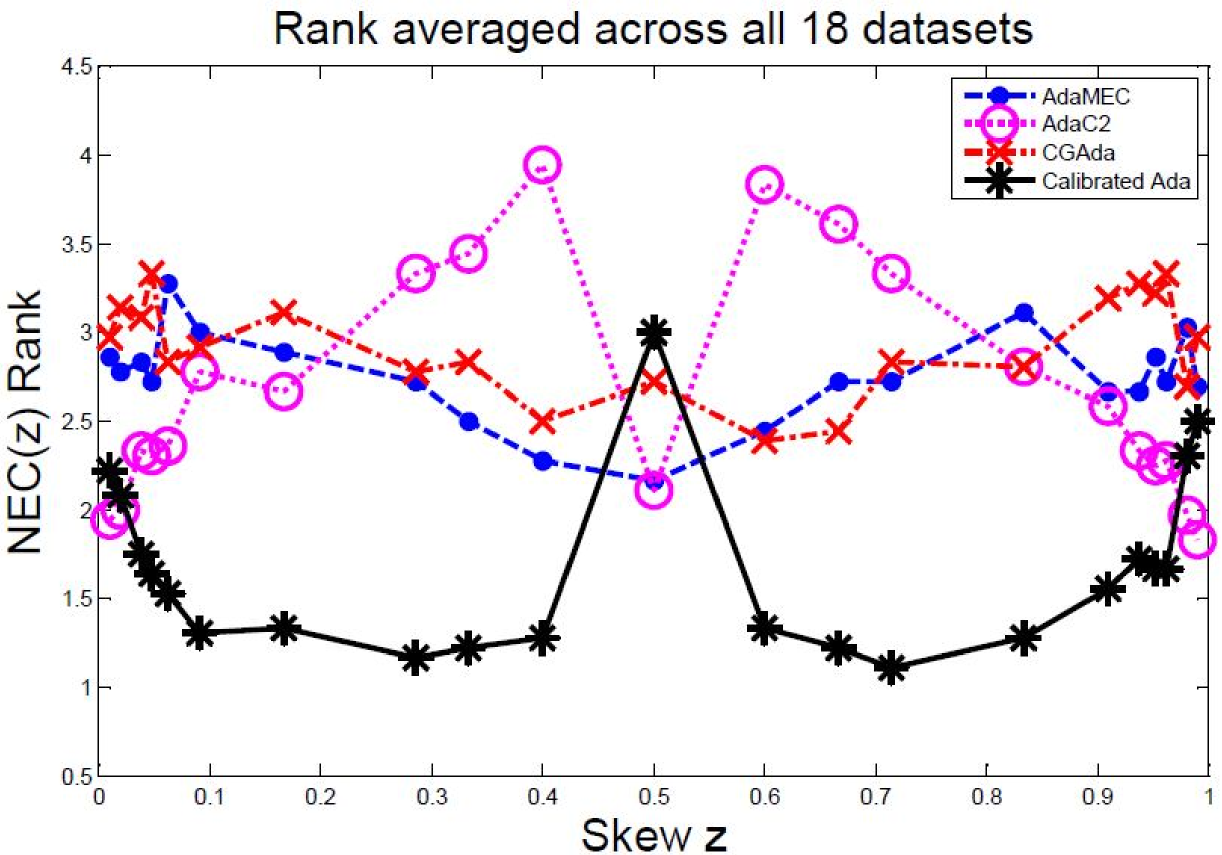
Empirical results (2)



Nemenyi test at the 0.05 level on the differences

Ada-Calibrated at least as good as best (no sig. diff.) for very low /high skew

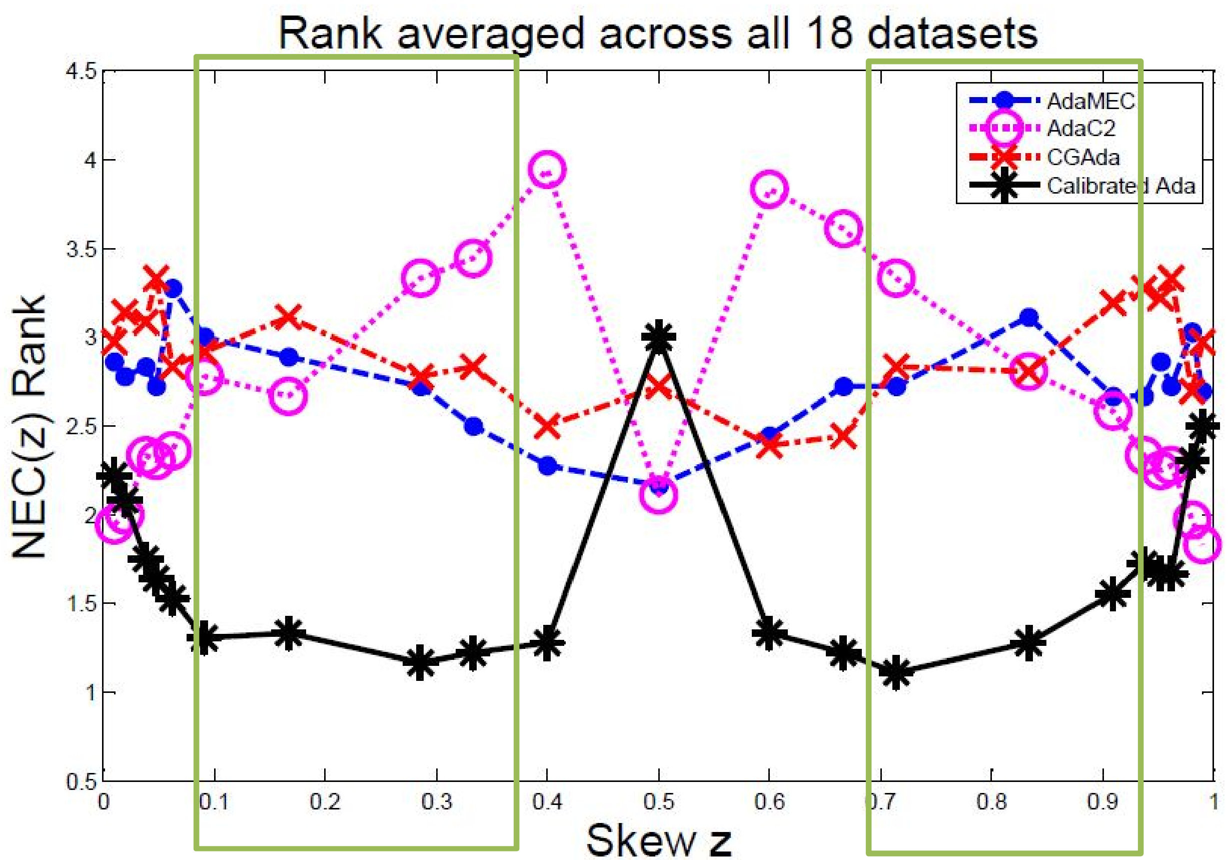
Empirical results (2)



Nemenyi test at the 0.05 level on the differences

Ada-Calibrated at least as good as best (no sig. diff.) for very low \high skew

Empirical results (2)



Nemenyi test at the 0.05 level on the differences

Ada-Calibrated at least as good as best (no sig. diff.) for very low \high skew

Ada-Calibrated superior to rest (sig. diff.) for medium skew

Conclusion

- Calibrating AdaBoost empirically **at least as good as best** among alternatives published 1998 - 2015

- Conceptual **simplicity**; no need for new algorithms, or hyperparameter setting

- **No need to retrain** if skew ratio changes in deployment

- Retains **theoretical guarantees** of AdaBoosty

- Sound **probabilistic / decision-theoretic motivation**

Thank you!

Additional Material

Boosting as a Product of Experts

AdaBoost: $$\hat{p}(y=1|\mathbf{x};F_M) = \frac{\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)}{\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)+\prod_{t=1}^{M}\hat{p}(y=-1|\mathbf{x};f_t)}$$

(Edakunni et al., 2011)

Boosting as a Product of Experts

AdaBoost: $$\hat{p}(y=1|\mathbf{x};F_M) = \frac{\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)}{\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)+\prod_{t=1}^{M}\hat{p}(y=-1|\mathbf{x};f_t)}$$

(Edakunni et al., 2011)

AdaMEC: $$\hat{p}(y=1|\mathbf{x};F_M) = \frac{c_{FN}\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)}{c_{FN}\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)+c_{FP}\prod_{t=1}^{M}\hat{p}(y=-1|\mathbf{x};f_t)}$$

Boosting as a Product of Experts

AdaBoost: $$\hat{p}(y=1|\mathbf{x};F_M) = \frac{\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)}{\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)+\prod_{t=1}^{M}\hat{p}(y=-1|\mathbf{x};f_t)}$$

(Edakunni et al., 2011)

AdaMEC: $$\hat{p}(y=1|\mathbf{x};F_M) = \frac{c_{FN}\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)}{c_{FN}\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)+c_{FP}\prod_{t=1}^{M}\hat{p}(y=-1|\mathbf{x};f_t)}$$

AdaC2: $$\hat{p}(y=1|\mathbf{x};F_M) = \frac{{c_{FN}}^M\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)}{{c_{FN}}^M\prod_{t=1}^{M}\hat{p}(y|\mathbf{x};f_t)+{c_{FP}}^M\prod_{t=1}^{M}\hat{p}(y=-1|\mathbf{x};f_t)}$$

$$\vdots$$

Boosting as a Product of Experts

AdaBoost: $$\hat{p}(y=1|\mathbf{x};F_M) = \frac{\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)}{\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)+\prod_{t=1}^{M}\hat{p}(y=-1|\mathbf{x};f_t)}$$

(Edakunni et al., 2011)

AdaMEC: $$\hat{p}(y=1|\mathbf{x};F_M) = \frac{c_{FN}\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)}{c_{FN}\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)+c_{FP}\prod_{t=1}^{M}\hat{p}(y=-1|\mathbf{x};f_t)}$$

AdaC2: $$\hat{p}(y=1|\mathbf{x};F_M) = \frac{{c_{FN}}^M\prod_{t=1}^{M}\hat{p}(y=1|\mathbf{x};f_t)}{{c_{FN}}^M\prod_{t=1}^{M}\hat{p}(y|\mathbf{x};f_t)+{c_{FP}}^M\prod_{t=1}^{M}\hat{p}(y=-1|\mathbf{x};f_t)}$$

$\vdots$

Calibration

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

- A classifier is **calibrated** if

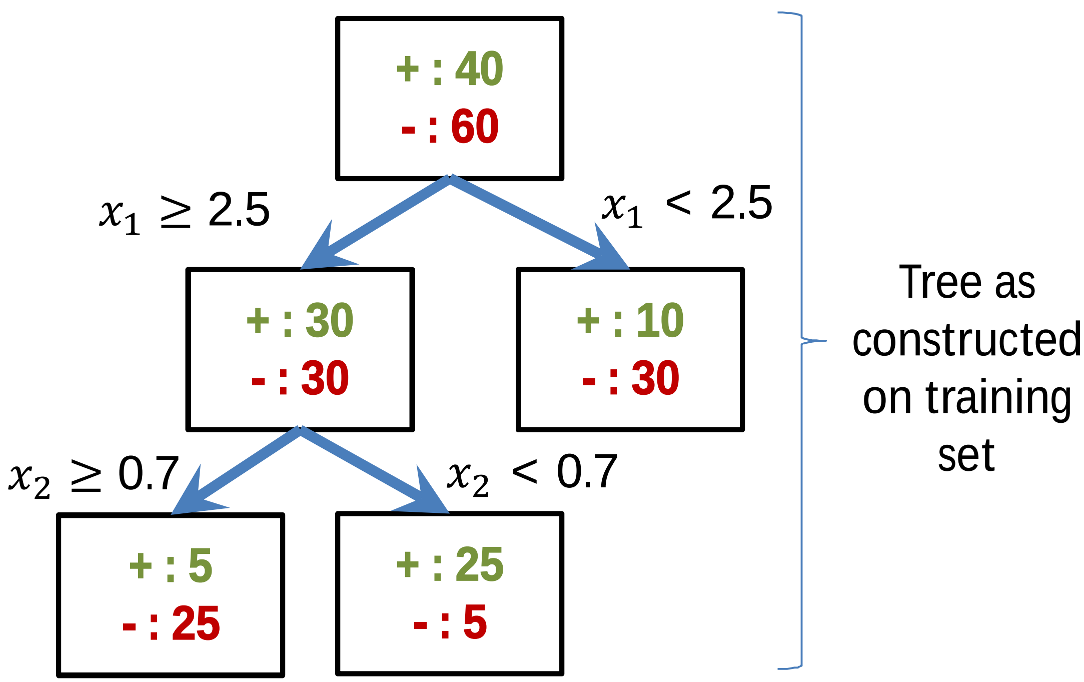
$$\hat{p}\ (y = 1|x) \triangleq \frac{N_{y=1,x}}{N_x} \rightarrow s(x), \text{ as } N \rightarrow \infty$$

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

- Calibration **can only improve** classification (asymptotically)

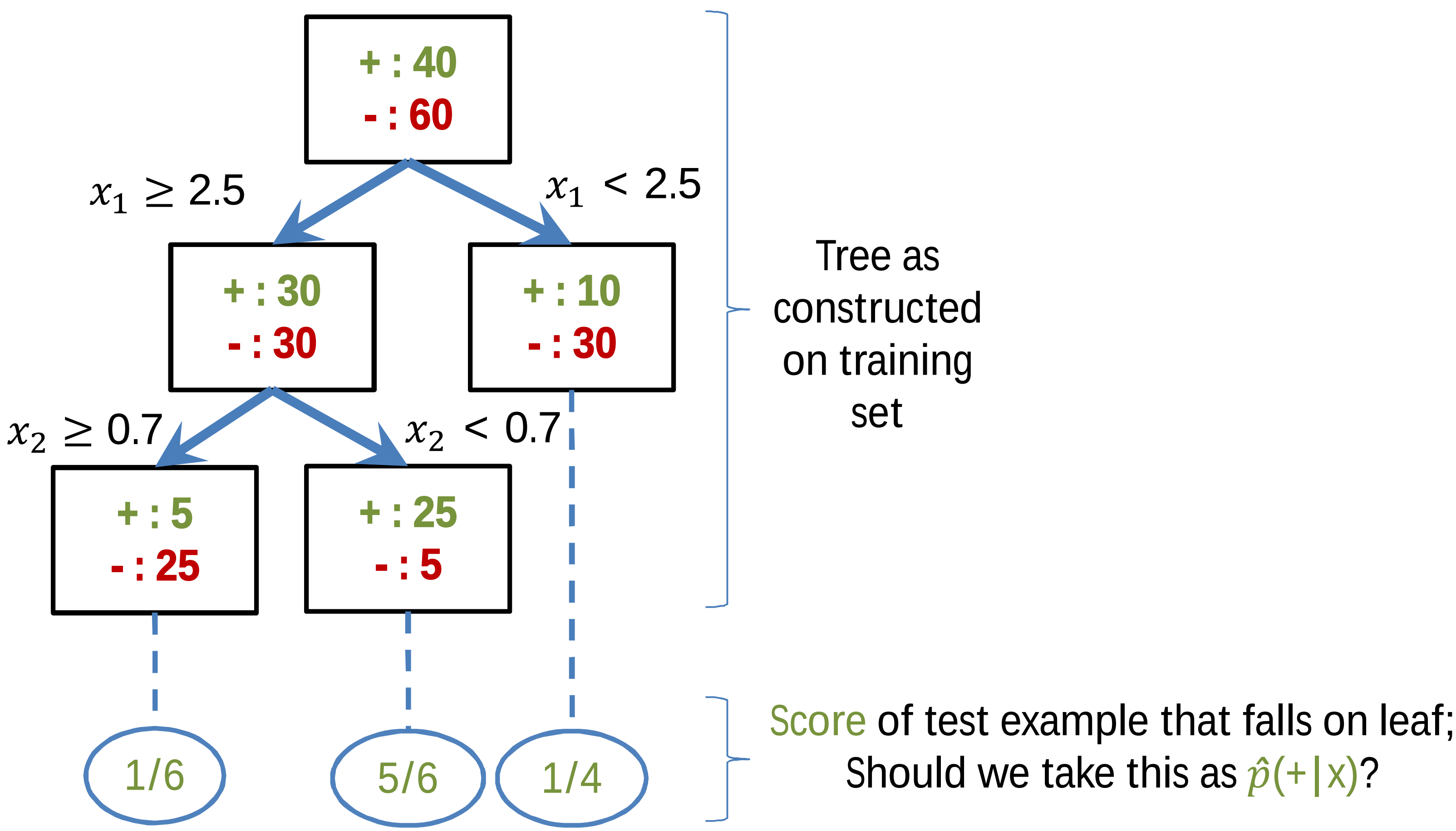
Probability estimation is not easy

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



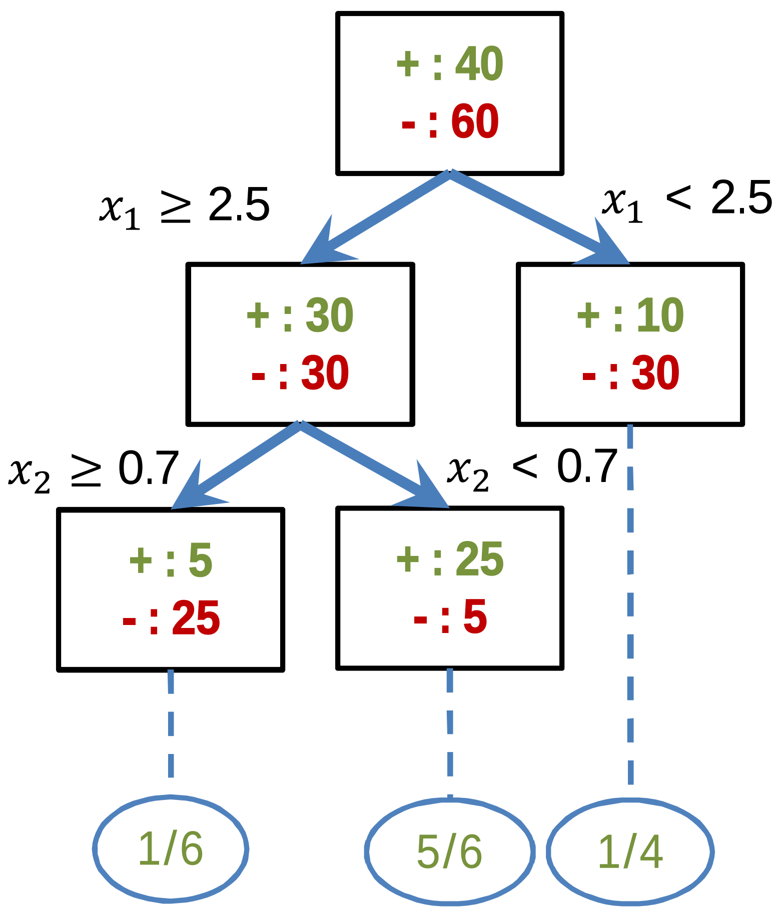
Probability estimation is not easy

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



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)