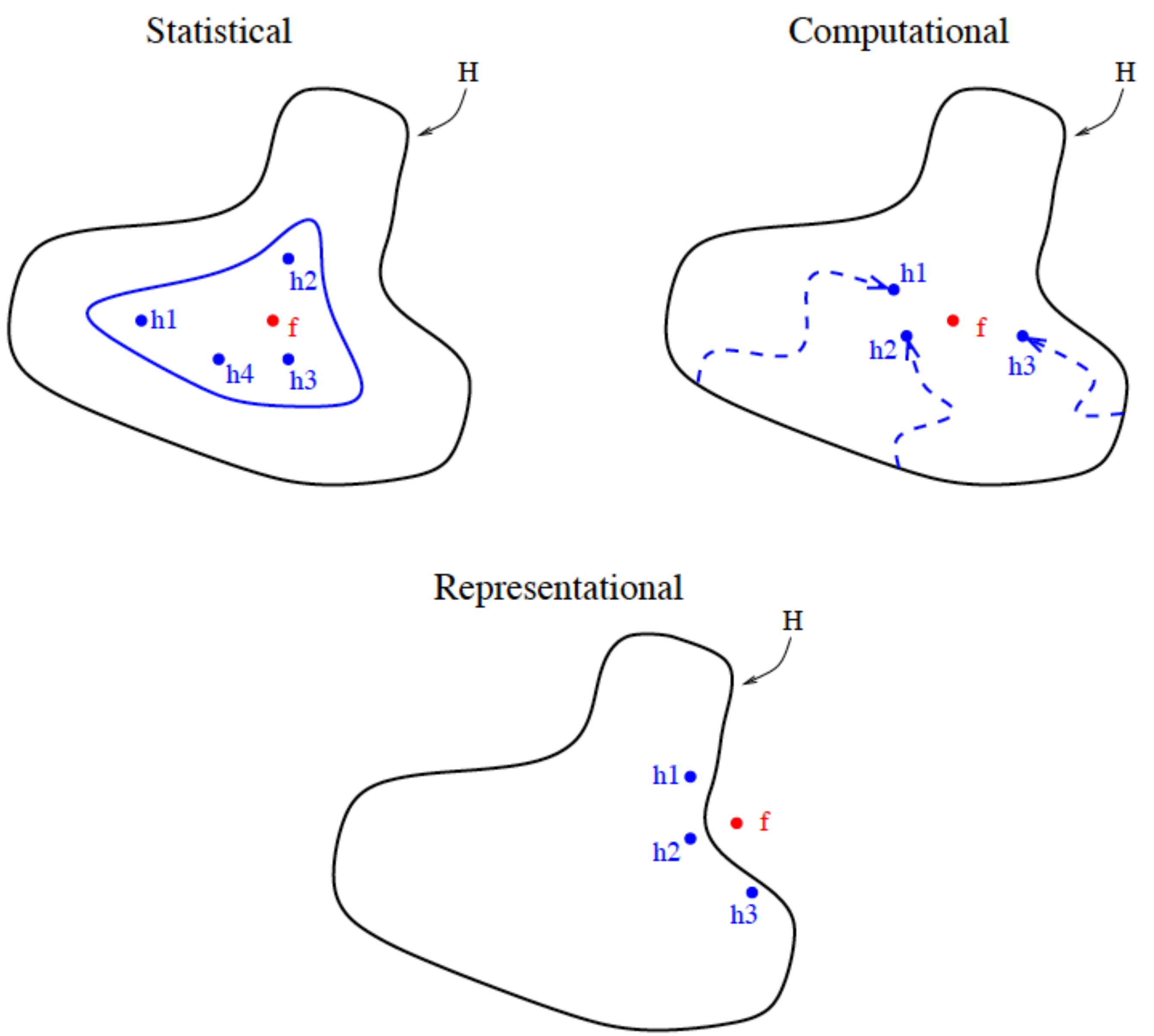
Ensemble Models

The idea

- The idea: construct a set of classifiers and combine them in certain way to classify new samples.
- The key to winning competitions
- To make it success: more accurate than each individual classifier
    - Need “accurate” (i.e. **better than random guessing**) and “diverse” (i.e. **make errors on different data samples**) classifiers
- The question: why does the idea of ensemble work?

- The three fundamental reasons
  - Statistical
    - Average out the votings to reduce the variance or bias
  - Computational
    - Combine the solutions obtained from local search
  - Representational
    - Possible to expand the space of the representation functions



Several Representatives

- General steps
  - 1. Producing a distribution of simple models on the subsets of the original data
  - 2. Combining the distribution into an “aggregated” model
- Three strategies
  - Bagging: decreasing variance
  - Boosting: decreasing bias
  - Stacking: improving the prediction power

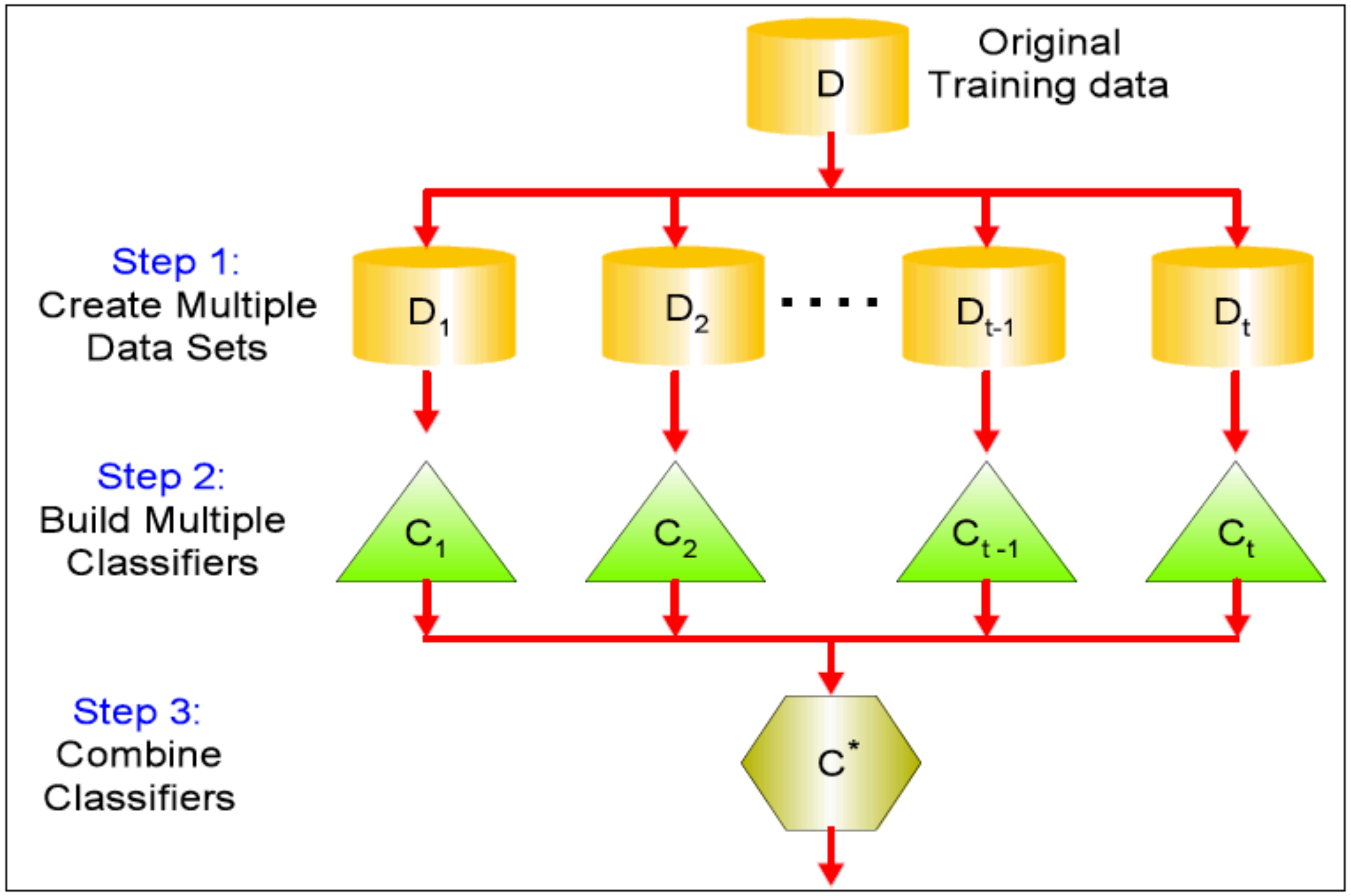
Bagging

- Abbreviated for **b**ootstrap **agg**regat**ing**

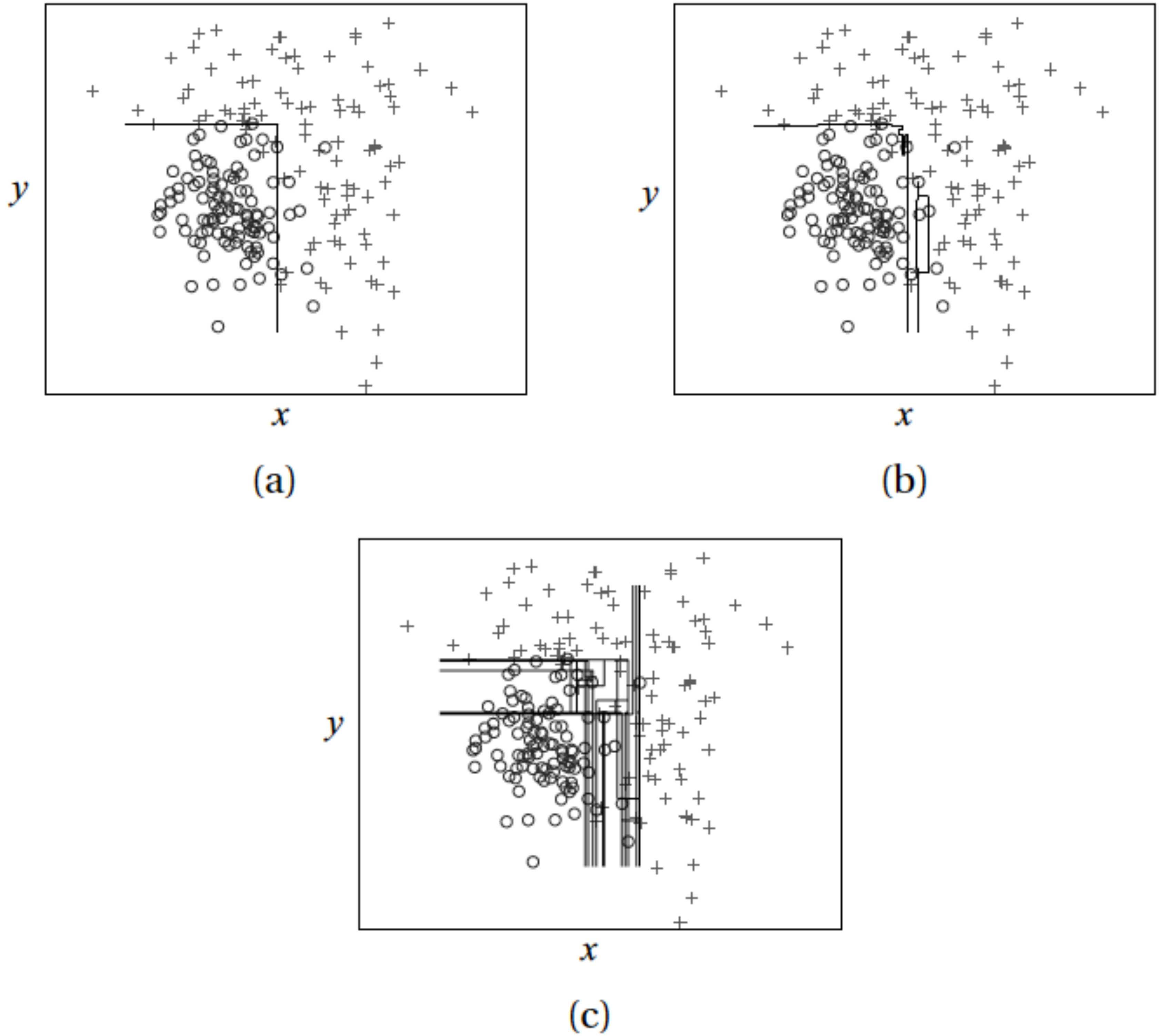
- Generating m new training sets by bootstrapping (i.e. sampling with replacement)

- Training each base classifier

- Combining them by averaging the output



Bagging 10 decision trees



Source: Ensemble methods: Foundation and Algorithms. Zhou (2012)

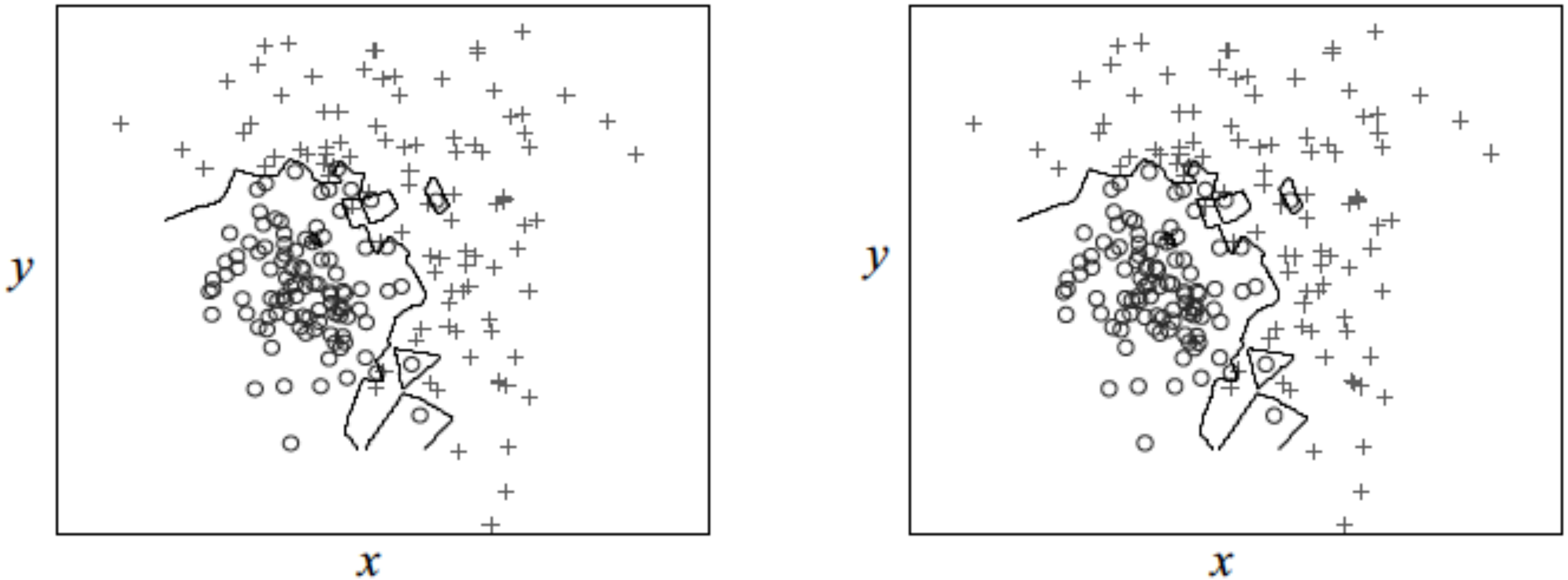
- Out-of-bag (oob) examples for estimating generalization error

$$H^{oob}(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} \sum_{t=1}^{T} \mathbb{I}(h_t(\boldsymbol{x}) = y) \cdot \mathbb{I}(\boldsymbol{x} \notin D_t)$$

$$err^{oob} = \frac{1}{|D|} \sum_{(\boldsymbol{x}, y) \in D} \mathbb{I}(H^{oob}(\boldsymbol{x}) \neq y)$$

- Bagging requires diverse/independent learners, and so **stable learners do not help**.
    - KNN is stable.
    - Decision trees are unstable learners w.r.t. bootstrap subsets, particularly unpruned trees. 😎

Bagging 1-NN

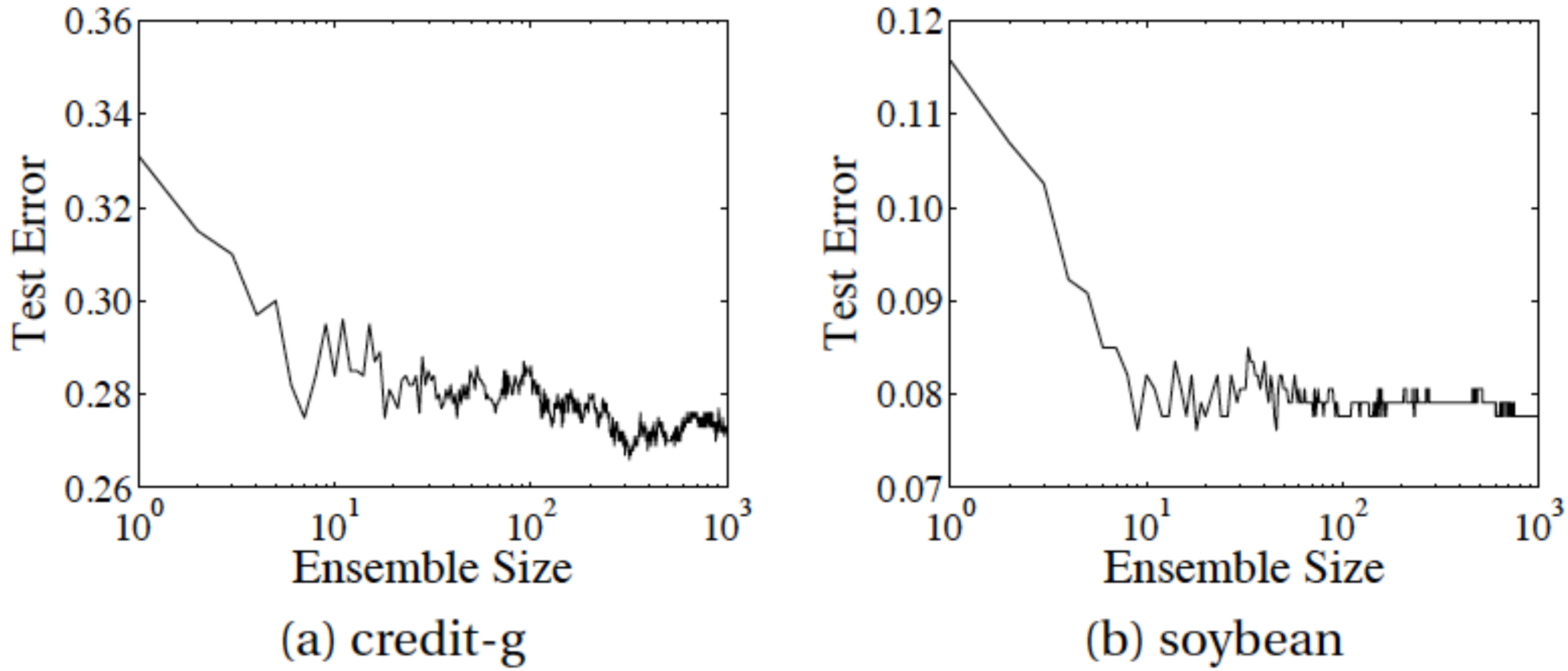


(a) (b)

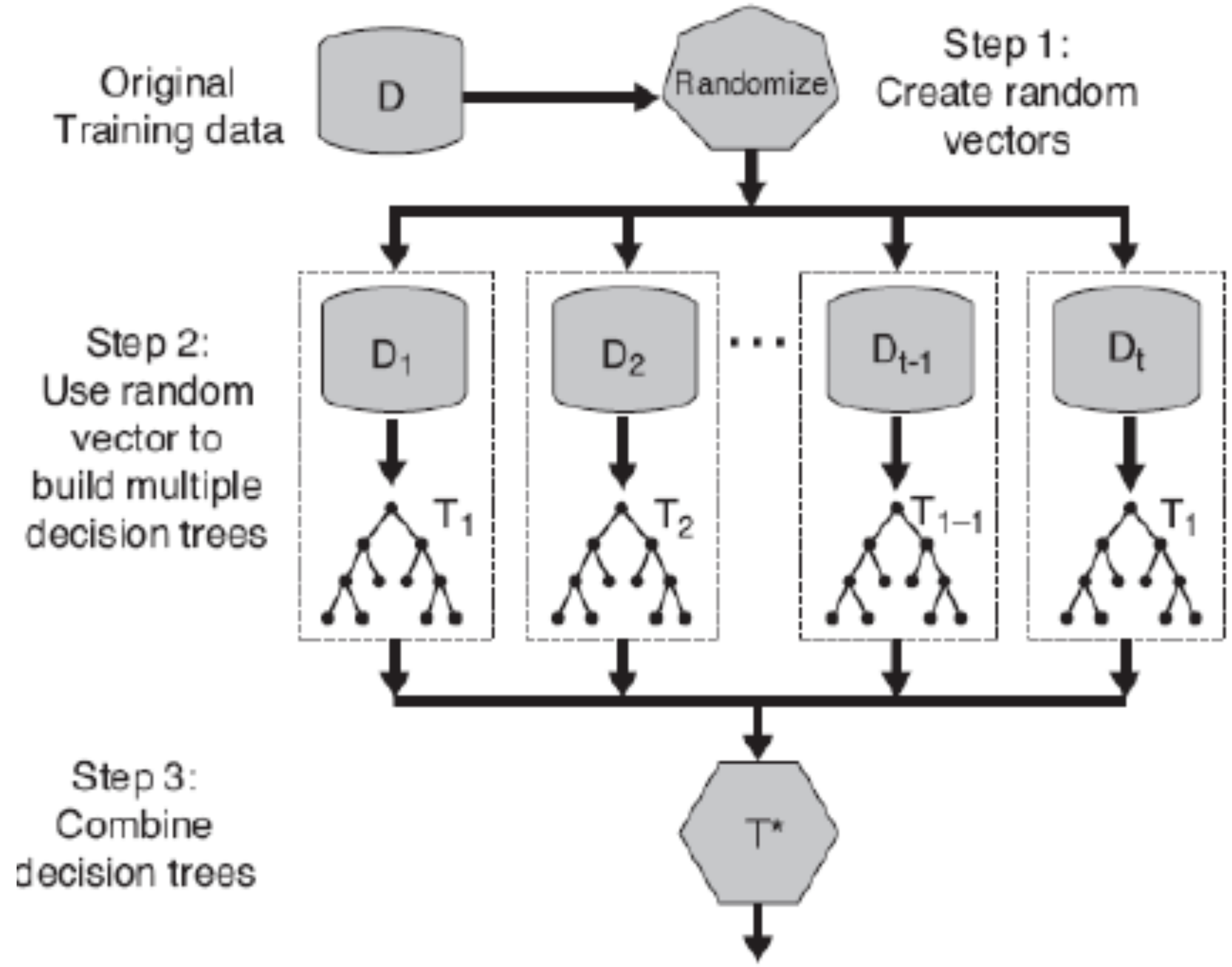
1-NN **Bagging 1-NN**

- With increasing ensemble size, the generalization converges.

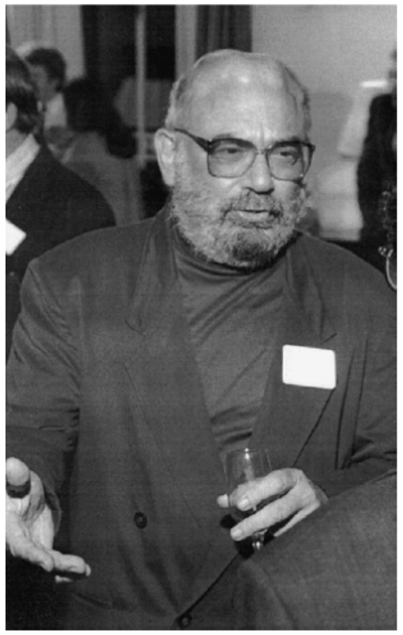
  - Why?



- An example: **random forest**
  - Ensemble method designed for decision trees.
  - Two sources of randomness.
    - Bagging: each tree is grown using a bootstrap sample of data
    - Random input vectors: at each node, best split is chosen from random m attributes instead of all attributes



Leo Breiman, 1928 - 2005



1954: PhD Berkeley (mathematics)

1960 -1967: UCLA (mathematics)

1969 -1982: Consultant

1982 - 1993 Berkeley (statistics)

1984 “Classification & Regression Trees” (with Friedman, Olshen, Stone)

1996 “Bagging”

2001 “Random Forests”

- Methods for growing the trees
  Fix a m <= p, at each node
  - Method 1
    - Choose d attributes randomly, compute their information gain, and choose the attribute with the largest gain to split.
  - Method 2
    - Compute a linear combination of L random attributes using weights generated from [-1, 1] randomly.
  - Method 3
    - Compute the information gain of all D attributes. Select the top d attributes by information gain. Randomly select one of d attributes as the splitting node.

Random Forest based on method 1

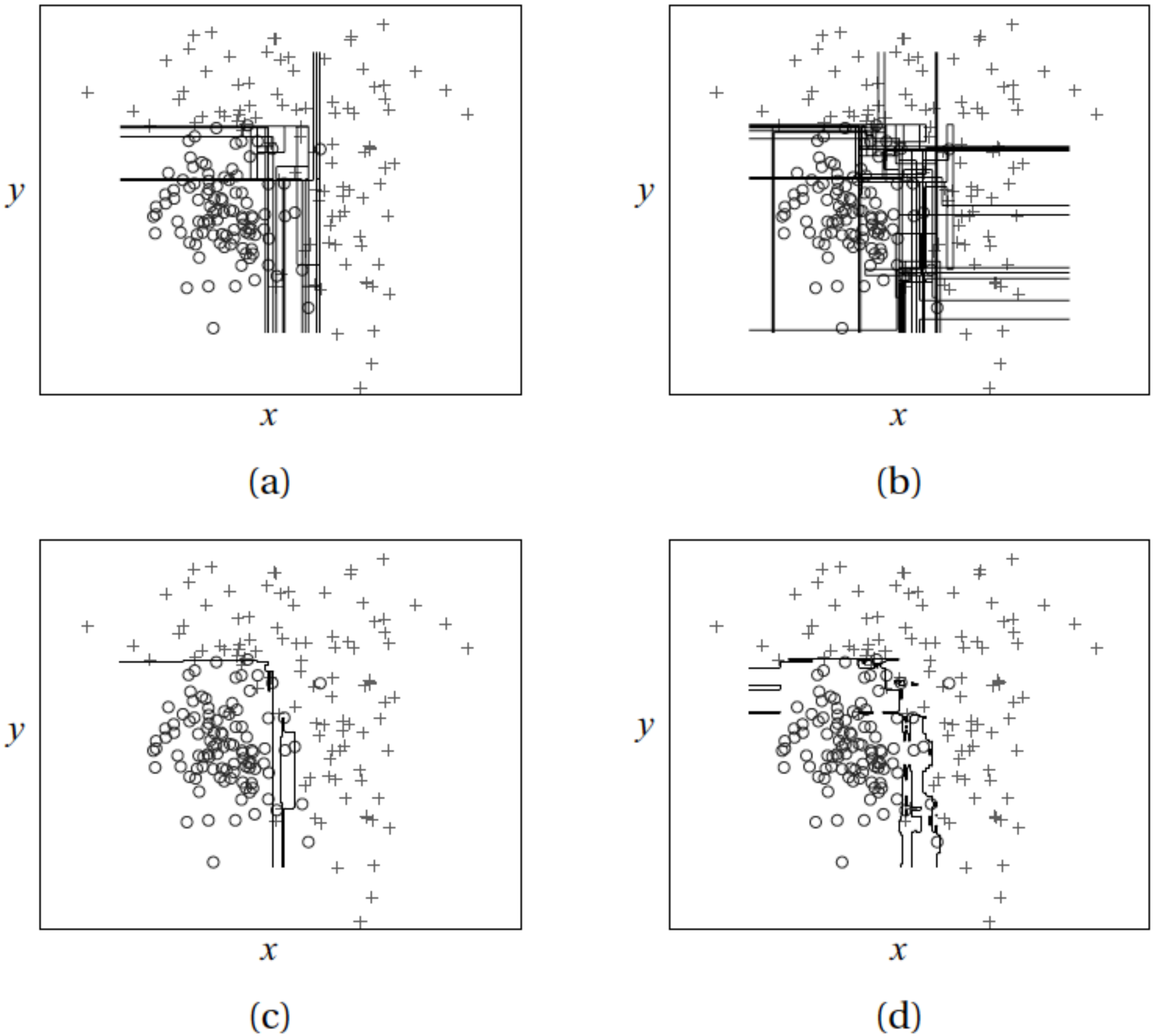
1. For $b = 1$ to B:
   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size N from the training data.
   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.
      i. Select m variables at random from the p variables.
      ii. Pick the best variable/split-point among the m.
      iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:

Regression: $\hat{f}_{\text{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^B T_b(x)$.

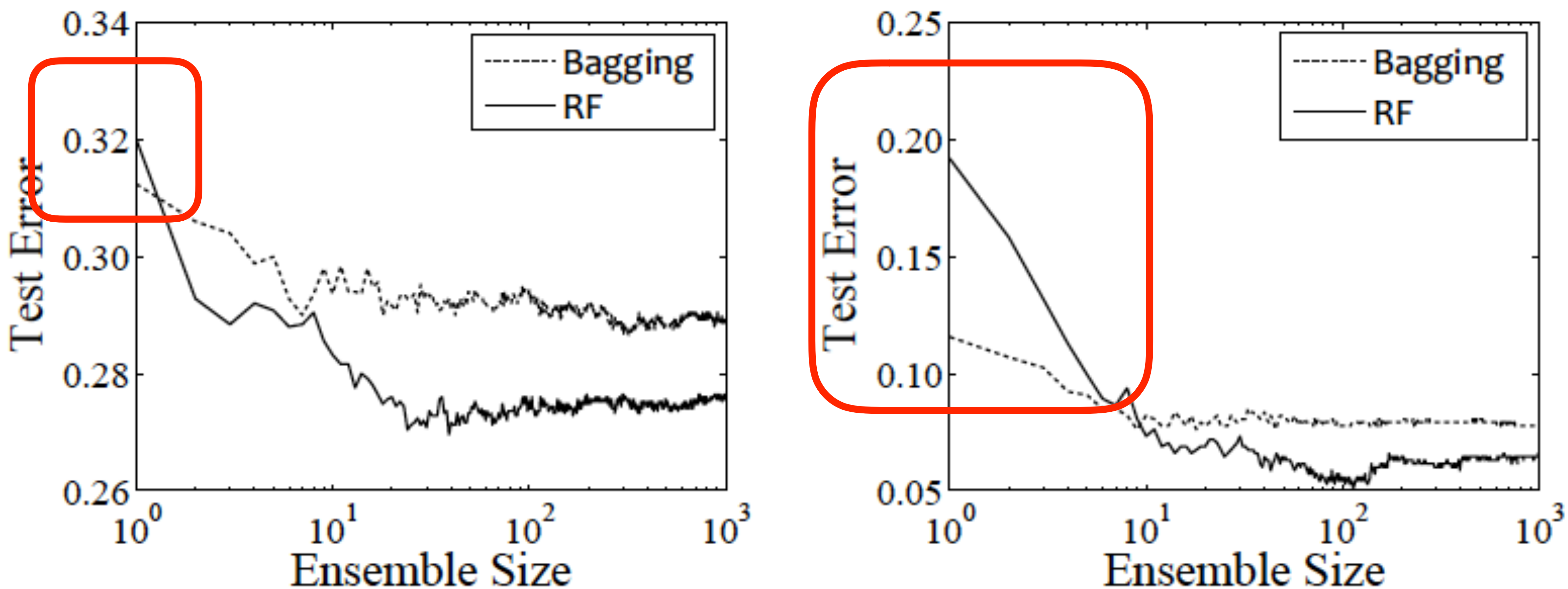
Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\text{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.

base classifiers of bagging **base classifiers of RF**



bagging **RF**

Convergence property



(a) credit-g (b) soybean

Some Remarks

- Bagging works through reducing the prediction variance.

- Some simple analysis

$$P(h_i(x) \neq f(x)) = \epsilon$$

H makes errors when at least half of the base classifiers make error

$$H(x) = \mathtt{sign}\left(\sum_{i=1}^{T} h_i(x)\right)$$

Assuming independence

$$P(H(x) \neq f(x)) = \sum_{k=0}^{\lfloor T/2 \rfloor} \binom{T}{k} (1-\epsilon)^k \epsilon^{T-k} \leq \exp\left(-\frac{1}{2} T (2\epsilon - 1)^2\right)$$

Hoeffding inequality

- Bagging is amenable to parallel processing.

Bagging can reduce variance,
can we decrease bias?

Boosting

- A family of algorithms that **turn weak learners to strong ones.**
- The idea
  - **Sequentially** train a set of classifiers and combine them
  - Late learners focus more on mistakes of the earlier learners

---

Input: Sample distribution $\mathcal{D}$;
Base learning algorithm $\mathfrak{L}$;
Number of learning rounds T.

Process:

1. $\mathcal{D}_1 = \mathcal{D}$. % Initialize distribution
2. **for** $t = 1, \ldots, T$:
3. $h_t = \mathfrak{L}(\mathcal{D}_t)$; % Train a weak learner from distribution $\mathcal{D}_t$
4. $\epsilon_t = P_{\boldsymbol{x} \sim D_t}(h_t(\boldsymbol{x}) \neq f(\boldsymbol{x}))$; % Evaluate the error of h_t
5. $\mathcal{D}_{t+1} = Adjust\_Distribution(\mathcal{D}_t, \epsilon_t)$
6. **end**

Output: $H(\boldsymbol{x}) = Combine\_Outputs(\{h_1(\boldsymbol{x}), \ldots, h_t(\boldsymbol{x})\})$

Zhou (2012)

- An instantiation of boosting: **AdaBoost** (Freund and Schapire 1997, Friedman et.al 2000)

Input: Data set $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
Base learning algorithm $\mathfrak{L}$;
Number of learning rounds T.

Process:
1. $\mathcal{D}_1(\boldsymbol{x}) = 1/m$. % Initialize the weight distribution
2. **for** $t = 1, \ldots, T$:
3. $h_t = \mathfrak{L}(D, \mathcal{D}_t)$; % Train a classifier h_t from D under distribution $\mathcal{D}_t$
4. $\epsilon_t = P_{\boldsymbol{x} \sim \mathcal{D}_t}(h_t(\boldsymbol{x}) \neq f(\boldsymbol{x}))$; % Evaluate the error of h_t
5. **if** $\epsilon_t > 0.5$ **then break**
6. $\alpha_t = \frac{1}{2} \ln \left(\frac{1-\epsilon_t}{\epsilon_t} \right)$; % Determine the weight of h_t
7. $$\mathcal{D}_{t+1}(\boldsymbol{x}) = \frac{\mathcal{D}_t(\boldsymbol{x})}{Z_t} \times \begin{cases} \exp(-\alpha_t) & \text{if } h_t(\boldsymbol{x}) = f(\boldsymbol{x}) \\ \exp(\alpha_t) & \text{if } h_t(\boldsymbol{x}) \neq f(\boldsymbol{x}) \end{cases}$$ **Re-weighting**
$$= \frac{\mathcal{D}_t(\boldsymbol{x}) \exp(-\alpha_t f(\boldsymbol{x}) h_t(\boldsymbol{x}))}{Z_t}$$ % Update the distribution, where
% Z_t is a normalization factor which
% enables $\mathcal{D}_{t+1}$ to be a distribution
8. **end**

Output: $H(\boldsymbol{x}) = \mathtt{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) \right)$

Zhou (2012)

- Minimizing exponential loss

$$\ell_{\exp}(h \mid \mathcal{D}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})h(\boldsymbol{x})}] \qquad H(\boldsymbol{x}) = \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x})$$

When the H is optimal, the following should hold.

$$\begin{aligned}\frac{\partial e^{-f(\boldsymbol{x})H(\boldsymbol{x})}}{\partial H(\boldsymbol{x})} &= -f(\boldsymbol{x})e^{-f(\boldsymbol{x})H(\boldsymbol{x})} \\ &= -e^{-H(\boldsymbol{x})}P(f(\boldsymbol{x}) = 1 \mid \boldsymbol{x}) + e^{H(\boldsymbol{x})}P(f(\boldsymbol{x}) = -1 \mid \boldsymbol{x}) \\ &= 0\,.\end{aligned}$$

$$\Downarrow$$

$$H(\boldsymbol{x}) = \frac{1}{2}\ln\frac{P(f(x) = 1 \mid \boldsymbol{x})}{P(f(x) = -1 \mid \boldsymbol{x})}$$

$$\texttt{sign}(H(\boldsymbol{x})) = \texttt{sign}\left(\frac{1}{2}\ln\frac{P(f(x)=1\mid\boldsymbol{x})}{P(f(x)=-1\mid\boldsymbol{x})}\right)$$
$$= \begin{cases} 1, & P(f(x)=1\mid\boldsymbol{x}) > P(f(x)=-1\mid\boldsymbol{x}) \\ -1, & P(f(x)=1\mid\boldsymbol{x}) < P(f(x)=-1\mid\boldsymbol{x}) \end{cases}$$
$$= \underset{y\in\{-1,1\}}{\arg\max}\, P(f(x)=y\mid\boldsymbol{x}),$$

Bayesian error rate

Construct from individual learner:

$$\ell_{\exp}(\alpha_t h_t \mid \mathcal{D}_t) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}[e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})}]$$
$$= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}\left[e^{-\alpha_t}\mathbb{I}(f(\boldsymbol{x})=h_t(\boldsymbol{x})) + e^{\alpha_t}\mathbb{I}(f(\boldsymbol{x})\neq h_t(\boldsymbol{x}))\right]$$
$$= e^{-\alpha_t}P_{\boldsymbol{x}\sim\mathcal{D}_t}(f(\boldsymbol{x})=h_t(\boldsymbol{x})) + e^{\alpha_t}P_{\boldsymbol{x}\sim\mathcal{D}_t}(f(\boldsymbol{x})\neq h_t(\boldsymbol{x}))$$
$$= e^{-\alpha_t}(1-\epsilon_t) + e^{\alpha_t}\epsilon_t\ ,$$

Optimal weight:

$$\frac{\partial \ell_{\exp}(\alpha_t h_t \mid \mathcal{D}_t)}{\partial \alpha_t} = -e^{-\alpha_t}(1-\epsilon_t) + e^{\alpha_t}\epsilon_t = 0 \longrightarrow \alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

Minimizing the loss:

$$\begin{aligned}\ell_{\exp}(H_{t-1}+h_t \mid \mathcal{D}) &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})(H_{t-1}(\boldsymbol{x})+h_t(\boldsymbol{x}))}] \\ &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}e^{-f(\boldsymbol{x})h_t(\boldsymbol{x})}]\end{aligned}$$

Taylor expansion:

$$\begin{aligned}\ell_{\exp}(H_{t-1}+h_t \mid \mathcal{D}) &\approx \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\left(1-f(\boldsymbol{x})h_t(\boldsymbol{x})+\frac{f(\boldsymbol{x})^2h_t(\boldsymbol{x})^2}{2}\right)\right] \\ &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\left(1-f(\boldsymbol{x})h_t(\boldsymbol{x})+\frac{1}{2}\right)\right]\end{aligned}$$

$$\begin{aligned}h_t(\boldsymbol{x}) &= \arg\min_h \ell_{\exp}(H_{t-1}+h \mid \mathcal{D}) \\ &= \arg\min_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\left(1-f(\boldsymbol{x})h(\boldsymbol{x})+\frac{1}{2}\right)\right] \\ &= \arg\max_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}f(\boldsymbol{x})h(\boldsymbol{x})\right] \\ &= \arg\max_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\frac{e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]}f(\boldsymbol{x})h(\boldsymbol{x})\right],\end{aligned}$$

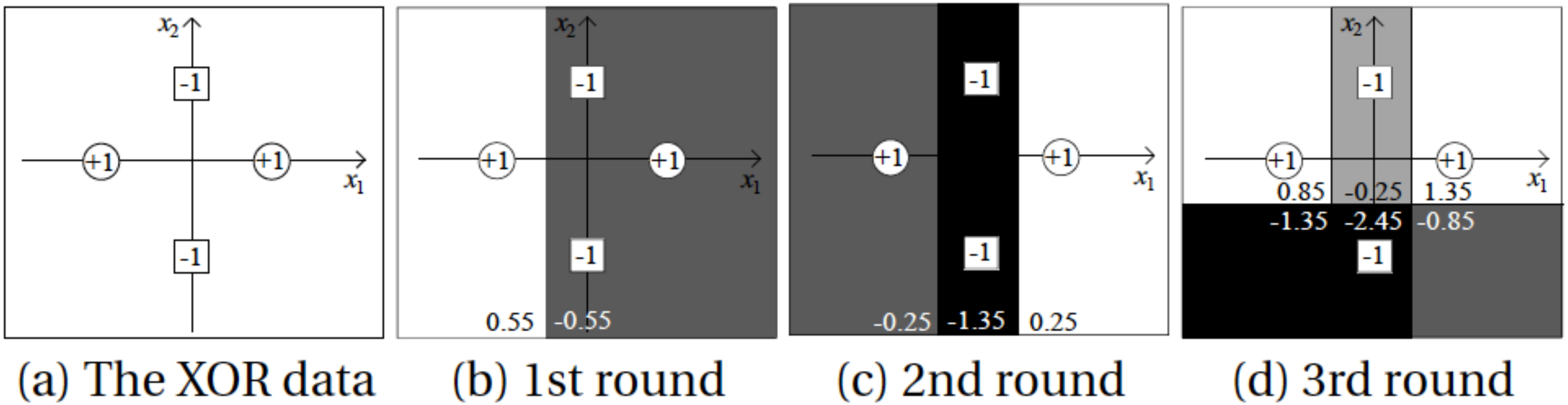
$$
\begin{aligned}
h_t(\boldsymbol{x}) &= \arg\max_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\frac{e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]} f(\boldsymbol{x})h(\boldsymbol{x})\right] \\
&= \arg\max_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}\left[f(\boldsymbol{x})h(\boldsymbol{x})\right] .
\end{aligned}
$$

$$
f(\boldsymbol{x})h_t(\boldsymbol{x}) = 1 - 2\mathbb{I}(f(\boldsymbol{x}) \neq h_t(\boldsymbol{x}))
$$

$$
h_t(\boldsymbol{x}) = \arg\min_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}\left[\mathbb{I}(f(\boldsymbol{x}) \neq h(\boldsymbol{x}))\right]
$$

$$
\begin{aligned}
\mathcal{D}_{t+1}(\boldsymbol{x}) &= \frac{\mathcal{D}(x)e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}]} \\
&= \frac{\mathcal{D}(\boldsymbol{x})e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}]} \\
&= \mathcal{D}_t(\boldsymbol{x}) \cdot e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})} \frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}]}
\end{aligned}
$$

- A simple example:



8 base learners

$$h_1(\boldsymbol{x}) = \begin{cases} +1, \text{ if } (x_1 > -0.5) \\ -1, \quad \text{otherwise} \end{cases} \quad h_2(\boldsymbol{x}) = \begin{cases} -1, \text{ if } (x_1 > -0.5) \\ +1, \quad \text{otherwise} \end{cases}$$

$$h_3(\boldsymbol{x}) = \begin{cases} +1, \text{ if } (x_1 > +0.5) \\ -1, \quad \text{otherwise} \end{cases} \quad h_4(\boldsymbol{x}) = \begin{cases} -1, \text{ if } (x_1 > +0.5) \\ +1, \quad \text{otherwise} \end{cases}$$

$$h_5(\boldsymbol{x}) = \begin{cases} +1, \text{ if } (x_2 > -0.5) \\ -1, \quad \text{otherwise} \end{cases} \quad h_6(\boldsymbol{x}) = \begin{cases} -1, \text{ if } (x_2 > -0.5) \\ +1, \quad \text{otherwise} \end{cases}$$

$$h_7(\boldsymbol{x}) = \begin{cases} +1, \text{ if } (x_2 > +0.5) \\ -1, \quad \text{otherwise} \end{cases} \quad h_8(\boldsymbol{x}) = \begin{cases} -1, \text{ if } (x_2 > +0.5) \\ +1, \quad \text{otherwise} \end{cases}$$

where x_1 and x_2 are the values of $\boldsymbol{x}$ at the first and the second dimension, respectively.

Rethinking AdaBoot

- Boosting is an additive model constructed sequentially.
- The exponential loss is an surrogate loss, an upper bound of 0-1 loss.

$$P(f(\boldsymbol{x}) = 1 \mid \boldsymbol{x}) = \frac{e^{H(\boldsymbol{x})}}{e^{H(\boldsymbol{x})} + e^{-H(\boldsymbol{x})}}$$

Log loss: $$\ell_{log}(h \mid \mathcal{D}) = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}} \left[\ln \left(1 + e^{-2f(\boldsymbol{x})h(\boldsymbol{x})} \right) \right]$$

- Optimization log loss with gradient descent with base regression models: LogitBoost
- Other variants: L2Boost, or more generally Gradient Boosting (functional gradient descent)
    - XGBoost Library

Input: Data set $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
Least square base learning algorithm $\mathfrak{L}$;
Number of learning rounds T.

Process:

1. $y_0(\boldsymbol{x}) = f(\boldsymbol{x})$. % Initialize target
2. $H_0(\boldsymbol{x}) = 0$. % Initialize function
3. **for** $t = 1, \ldots, T$:
4. $\quad p_t(\boldsymbol{x}) = \frac{1}{1+e^{-2H_{t-1}(\boldsymbol{x})}}$; %Calculate probability
5. $\quad y_t(\boldsymbol{x}) = \frac{y_{t-1}(\boldsymbol{x}) - p_t(\boldsymbol{x})}{p_t(\boldsymbol{x})(1-p_t(\boldsymbol{x}))}$; % Update target
6. $\quad \mathcal{D}_t(\boldsymbol{x}) = p_t(\boldsymbol{x})(1 - p_t(\boldsymbol{x}))$; % Update weight
7. $\quad h_t = \mathfrak{L}(D, y_t, \mathcal{D}_t)$; % Train a least square classifier h_t to fit y_t in data set D under distribution $\mathcal{D}_t$
8. $\quad H_t(\boldsymbol{x}) = H_{t-1}(\boldsymbol{x}) + \frac{1}{2} h_t(\boldsymbol{x})$; %Update combined classifier
9. **end**

Output: $H(\boldsymbol{x}) = \mathtt{sign}\left(\sum_{t=1}^{T} h_t(\boldsymbol{x})\right)$

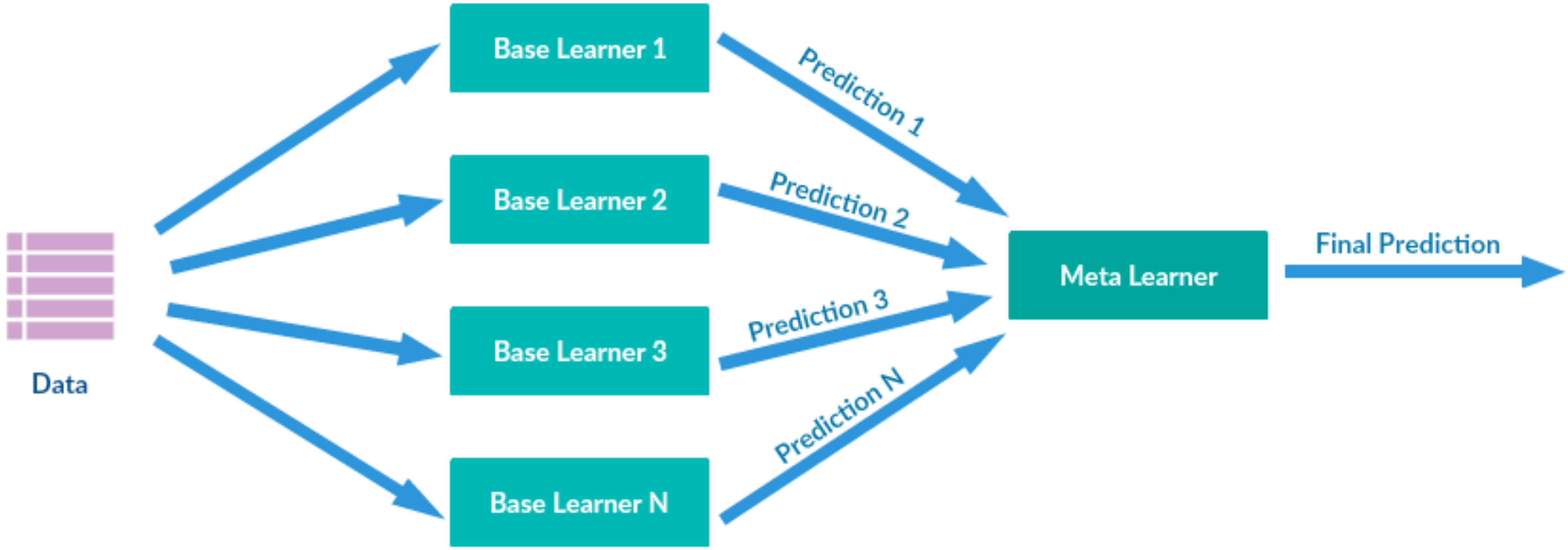
LogitBoost

Stacking

- Meta-learning methods: learning to combine (Wolpert 1992, Smyth and Wolpert 1998)

- A generalization of many ensemble methods

- The idea: the output of the base learners (the first-level learners) are used as the input features of the meta-learners (the second-level learners).



Input: Data set $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
First-level learning algorithms $\mathfrak{L}_1, \ldots, \mathfrak{L}_T$;
Second-level learning algorithm $\mathfrak{L}$.

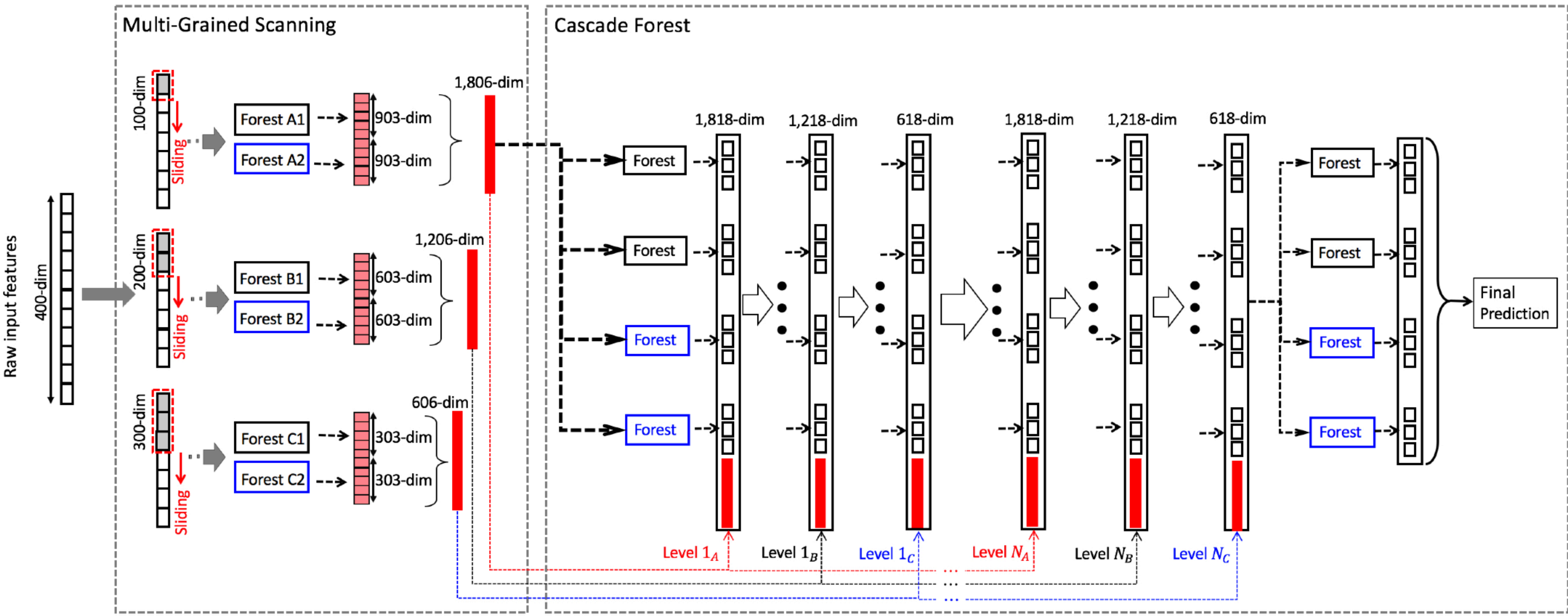
Process:

for $t = 1, \ldots, T$: % Train a first-level learner by applying the
  $h_t = \mathfrak{L}_t(D)$; % first-level learning algorithm $\mathfrak{L}_t$
end
$D' = \emptyset$; % Generate a new data set
for $i = 1, \ldots, m$:
  for $t = 1, \ldots, T$:
    $z_{it} = h_t(\boldsymbol{x}_i)$;
  end
  $D' = D' \cup ((z_{i1}, \ldots, z_{iT}), y_i)$;
end
$h' = \mathfrak{L}(D')$; % Train the second-level learner h' by applying
% the second-level learning algorithm $\mathfrak{L}$ to the
% new data set $\mathcal{D}'$.

Output: $H(\boldsymbol{x}) = h'(h_1(\boldsymbol{x}), \ldots, h_T(\boldsymbol{x}))$

A general procedure for stacking

- A recent example: Deep Forest (Zhou and Feng 2017)



- More topics of Boosting (for further study)
  - How to extend to multi-class cases?
  - Noise sensitivity (why sensitive?)
  - Generalization bound

Summary

- Ensemble models: combining different learners in various ways
  - Bagging: variance reduction, parellel
    - Random forest
  - Boosting: bias reduction, sequentially
    - AdaBoost
  - Stacking: learn how to combine
- Key: how to construct diverse base learners by introducing various types of randomness
- Typically an important trick to win competitions 😎
- Generalization error analysis

Exercises

- Derive how LogitBoost works.

- Readings

  - Generalization error analysis of AdaBoost
    - Freund, Yoav, and Robert E. Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting." *Journal of computer and system sciences* 55.1 (1997): 119-139.

    - Schapire, Robert E., et al. "Boosting the margin: A new explanation for the effectiveness of voting methods." *The annals of statistics* 26.5 (1998): 1651-1686.