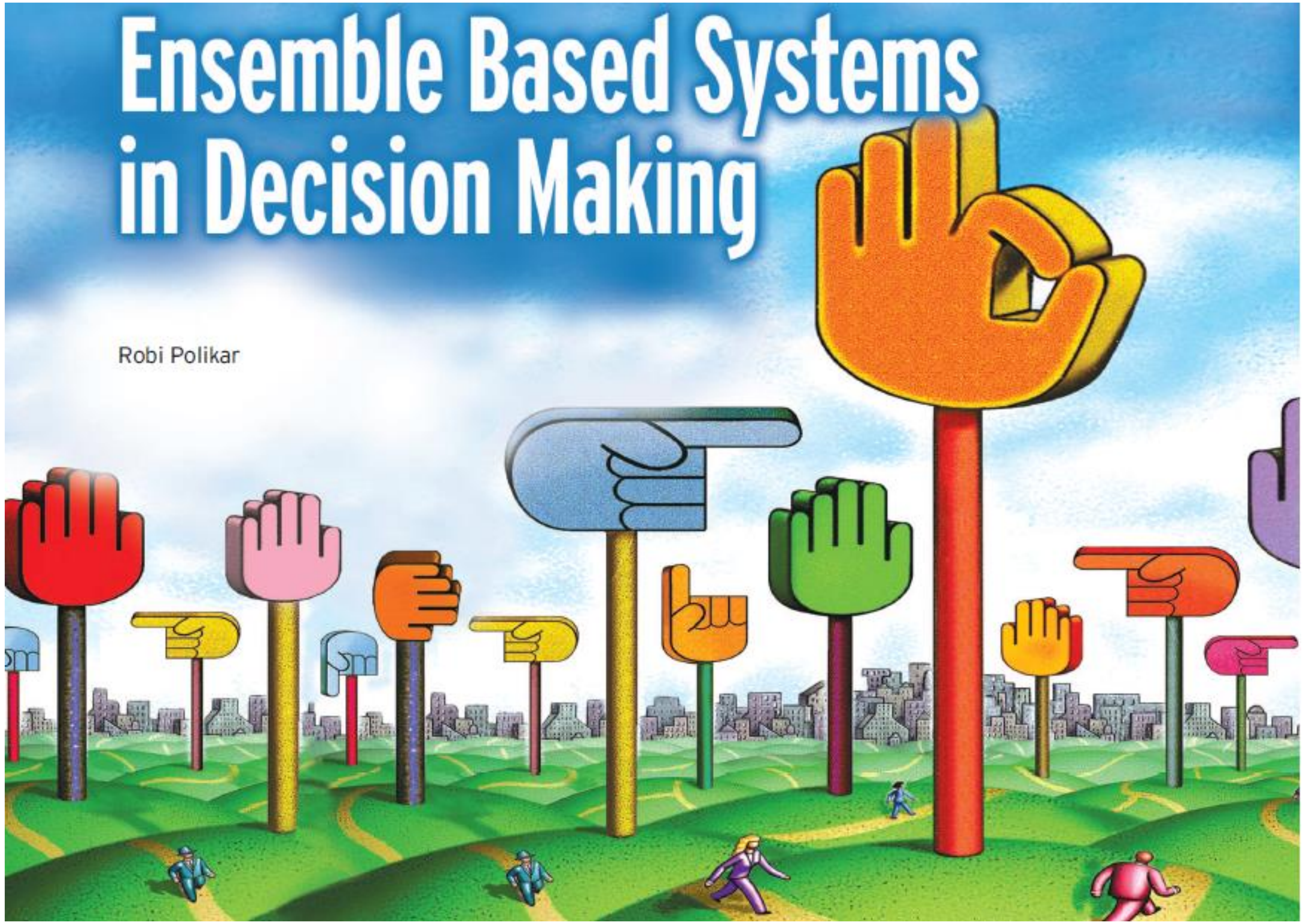
ENSEMBLE LEARNING

Reporter: Yunfei WANG
Email: yunfeiwang@hust.edu.cn

What's Ensemble Learning

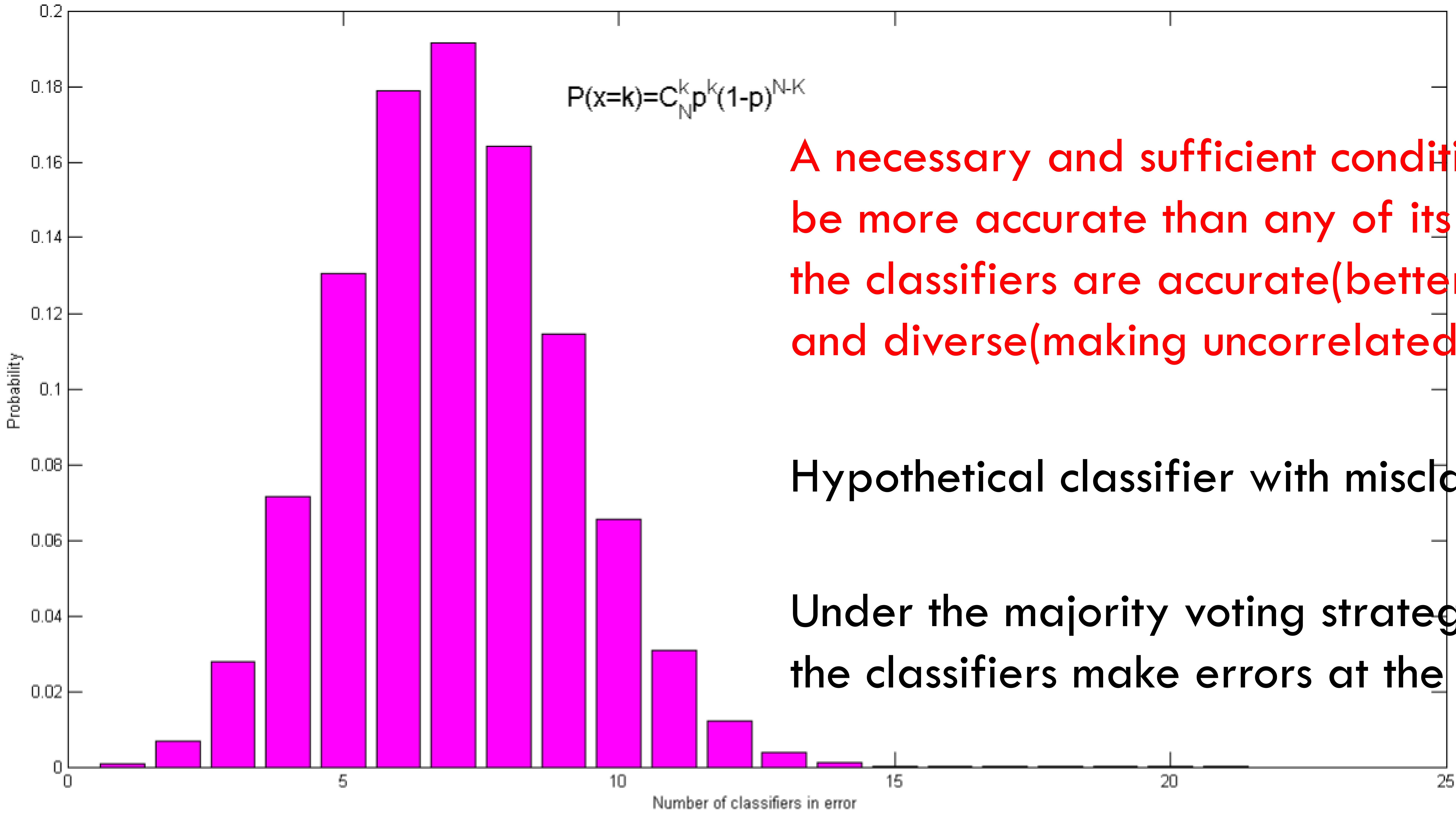


Ensemble learning is the process by which multiple modes are strategically generated and combined to solve a particular with better performance.

Applications

- Model selection
- Tasks with missing features
- Data fusion
- Incremental learning

Why does Ensemble Learning works?



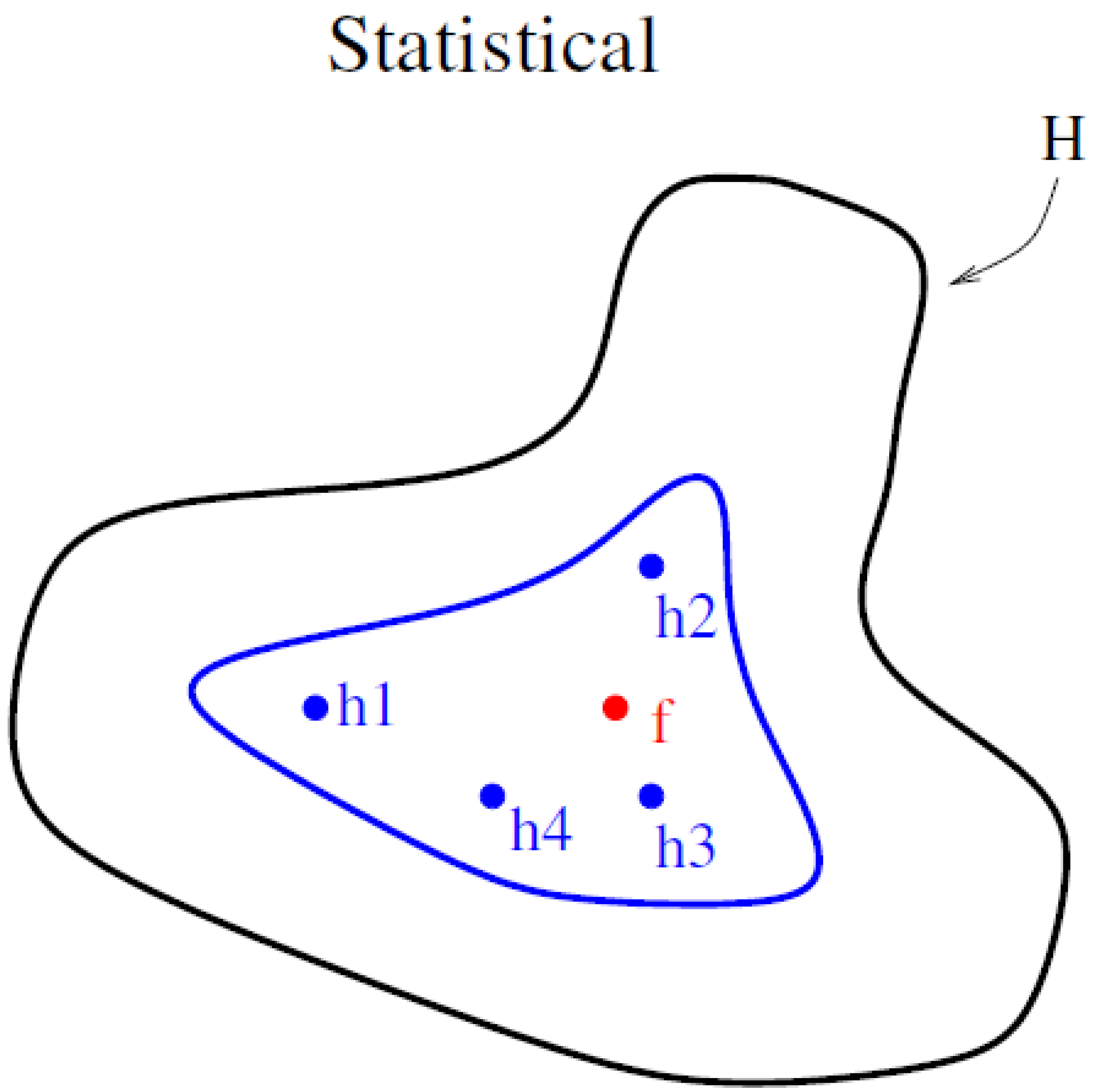
A necessary and sufficient condition for an ensemble to be more accurate than any of its individual members is if the classifiers are accurate(better than random guessing) and diverse(making uncorrelated errors).[5]

Hypothetical classifier with misclassification rate p=0.3

Under the majority voting strategy, it's not likely for half of the classifiers make errors at the same time.

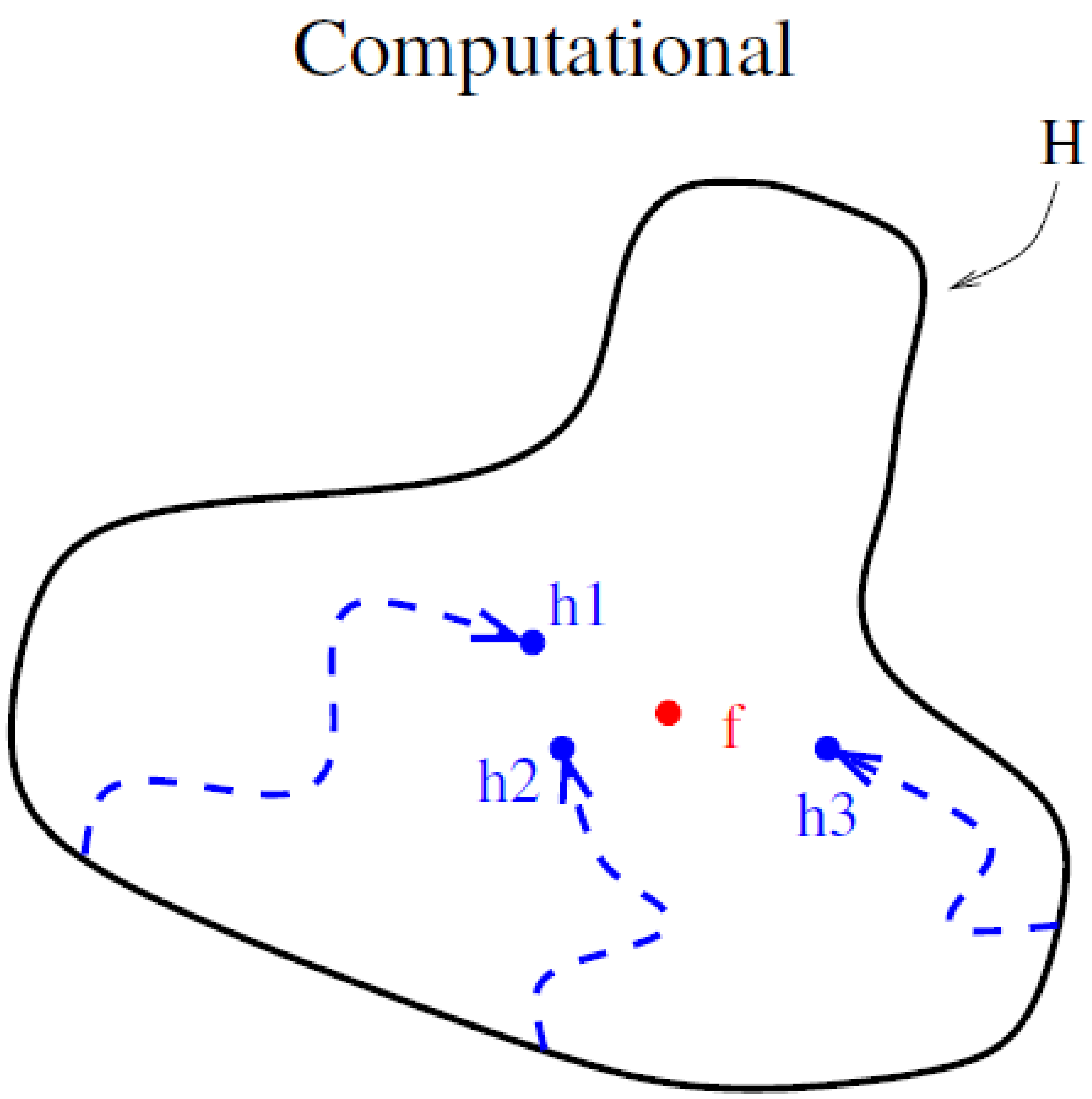
Why does Ensemble Learning works?[4,5]

Statistical Error



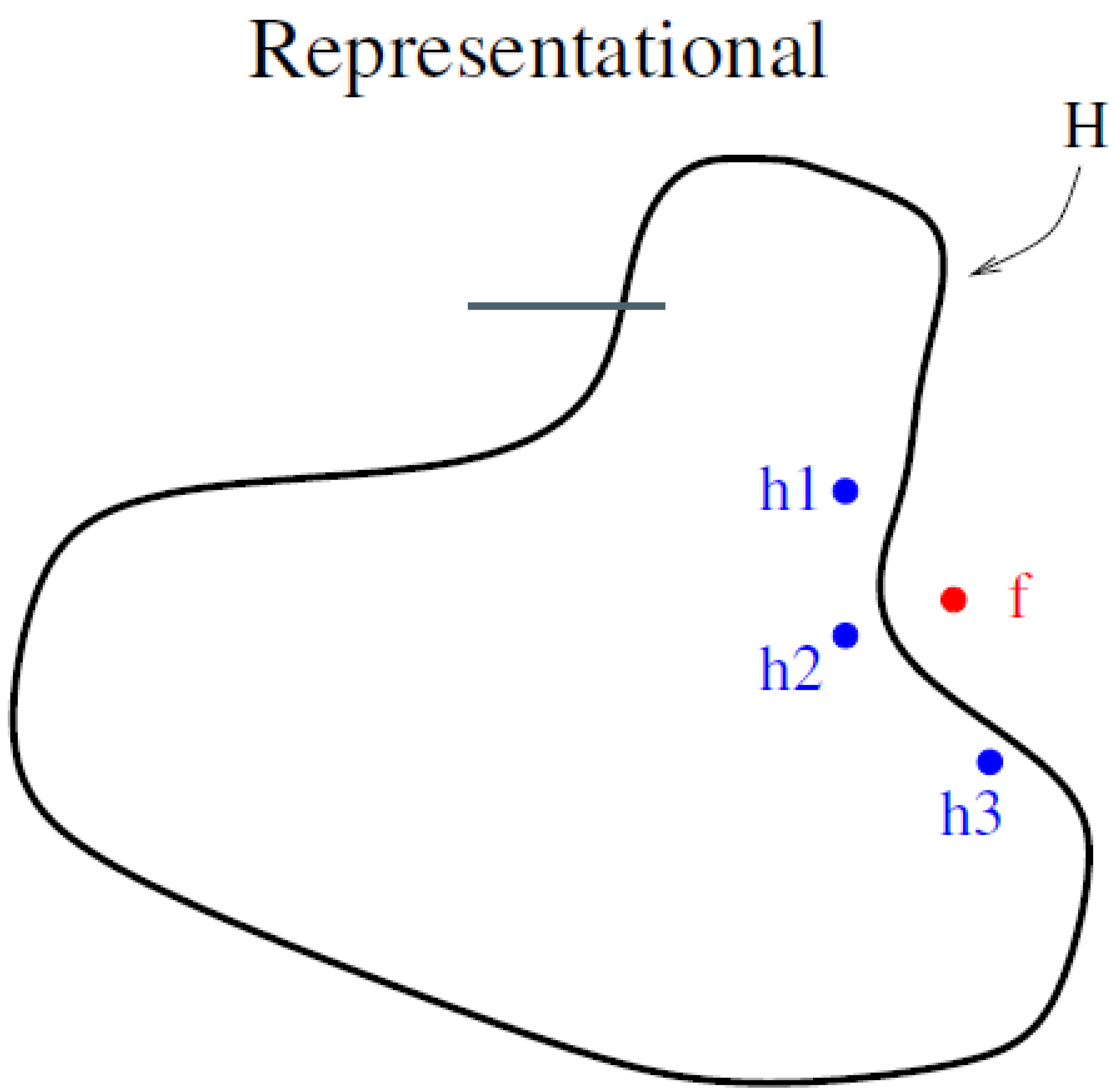
Combination of several hypothesizes produce an approximation closer to the true function.

Optimization Error



For hypothesis that are complex and sensitive to training set, ensemble outputs a more accurate hypothesis.

Model Error



Ensembles can produce an hypothesis outside the hypothesis set, expanding the representation abilities.

Diversity: Cornerstone of Ensemble Systems

- Models making different errors are complementary, therefore the combination of them can reduce the total error.
- How to achieve diversity?[1]

① Sample training data subsets randomly with replacement;

② Sample training data subsets randomly without replacement;

③ Set different parameters for the same type of different models;

④ Combine different type of models for diversity;

⑤ Different features contribute to diversity.

K-Folder Data Splitting

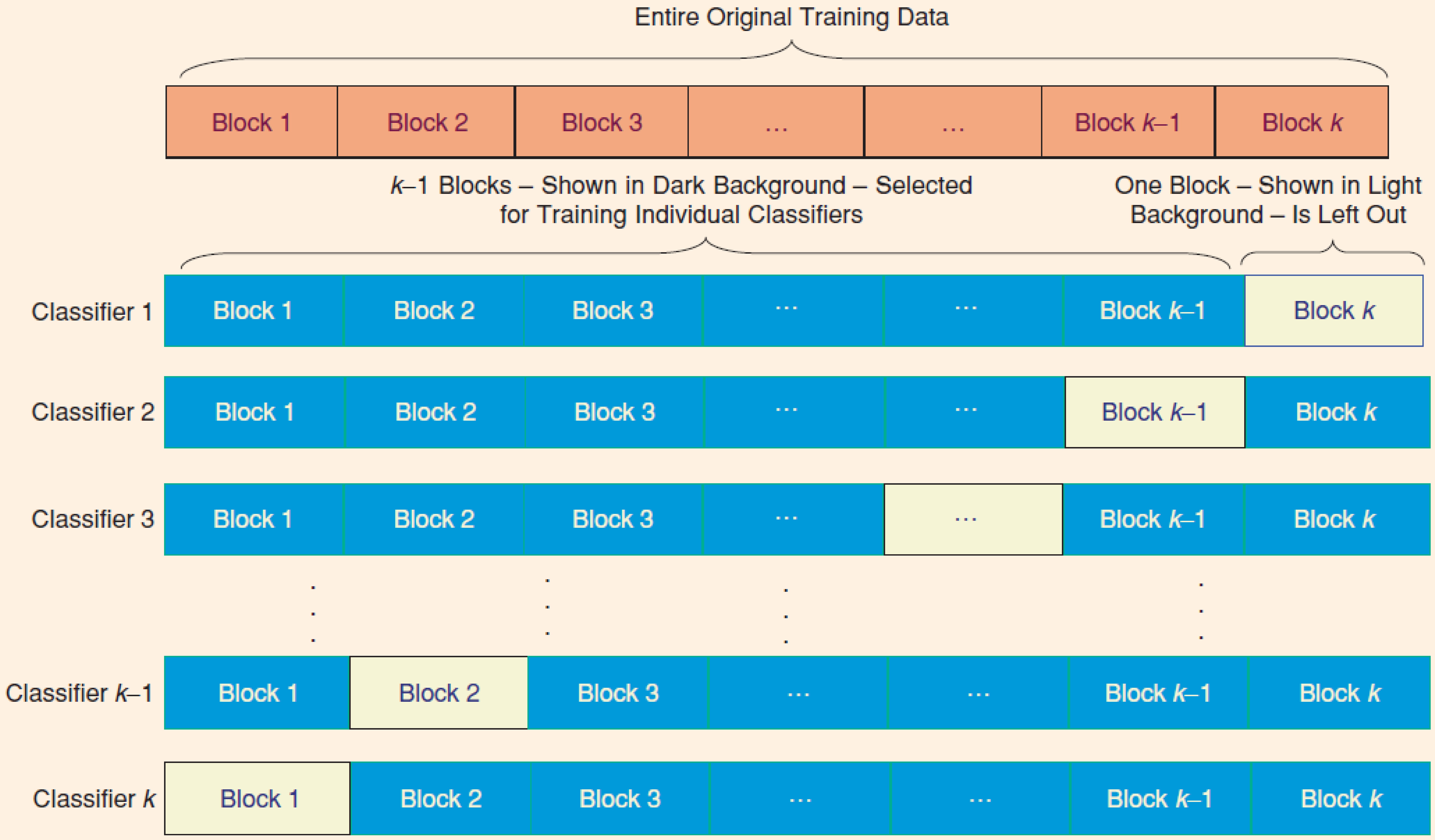
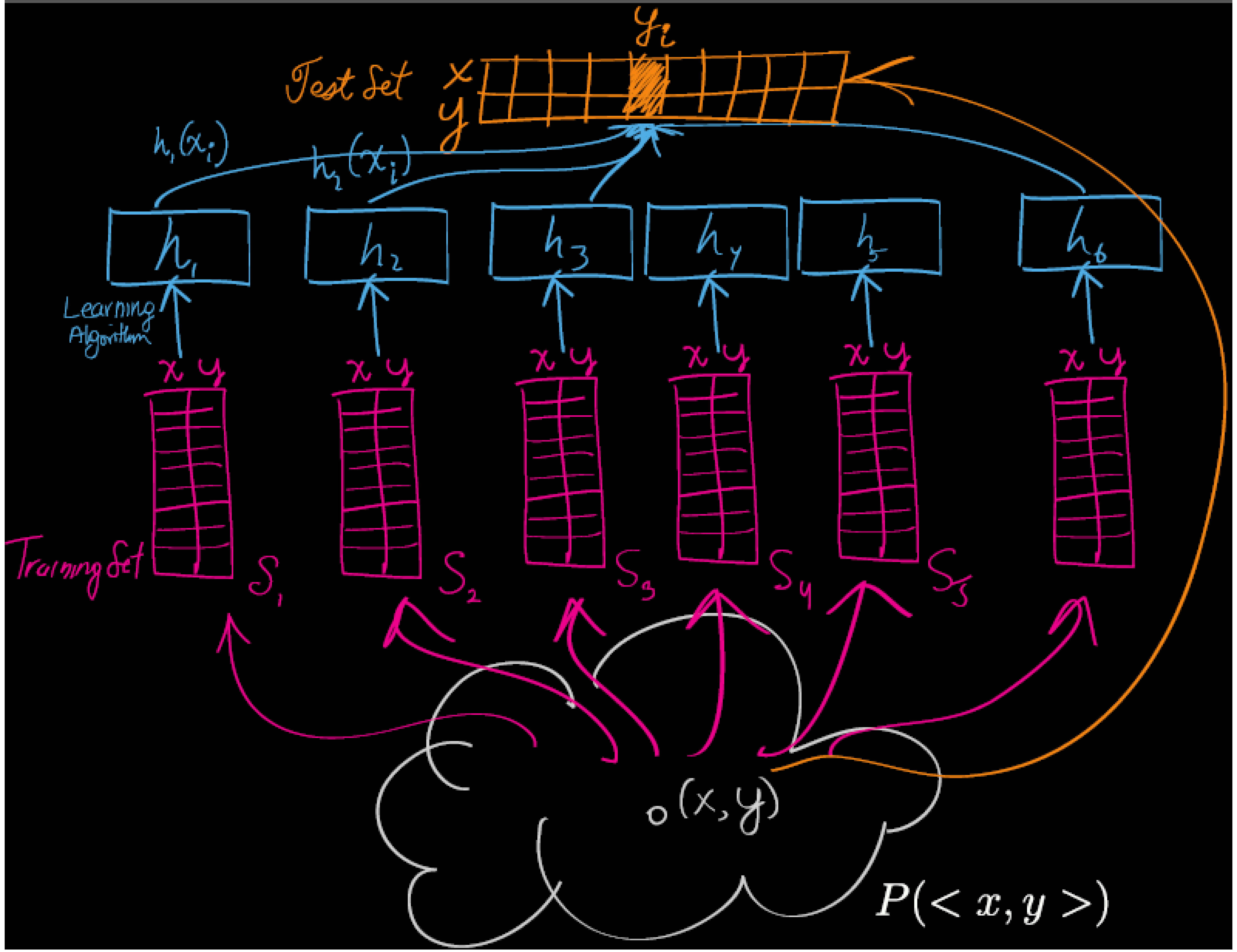


Figure 4. *k*-fold data splitting for generating different, but overlapping, training datasets.

Bagging



Appealing when training data is limited!

Draw training data subsets randomly with replacement from the entire training data

Large portions of samples(75%-100%) are drawn into each subset.

Train a different model of the same type on each training data subset

Unstable modes(Neural Networks/Decision Trees) can increase diversity for small perturbations in training set.

Combine models by averaging for regression or voting for classification.

Bagging-Algorithm

(Breiman, 1996)

Bagging (Boostrap aggregating).

$\textsc{Bagging}(S = ((x_1, y_1), \ldots, (x_m, y_m)))$

for $t \leftarrow 1$ **to** T **do**

$\quad S_t \leftarrow \textsc{Bootstrap}(S)$ $\triangleright$ i.i.d. sampling with replacement from S.

$\quad h_t \leftarrow \textsc{TrainClassifier}(S_t)$

return $h_S = x \mapsto \textsc{MajorityVote}((h_1(x), \ldots, h_T(x)))$

Ensemble :

$$h_S(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Bagging : Special case where we fix:

$\alpha_t = 1$ and $h_t = \mathbb{L}(S_t)$ *

* $\mathbb{L}$ is some learning algorithm

S_t is a training set drawn from distribution $P(< x, y >)$

Generalization Error

Classification :

$$\epsilon_{test} = \frac{1}{n}\sum_{i}^{n} \text{Zero-One-Loss}(y_i, h(x_i))$$

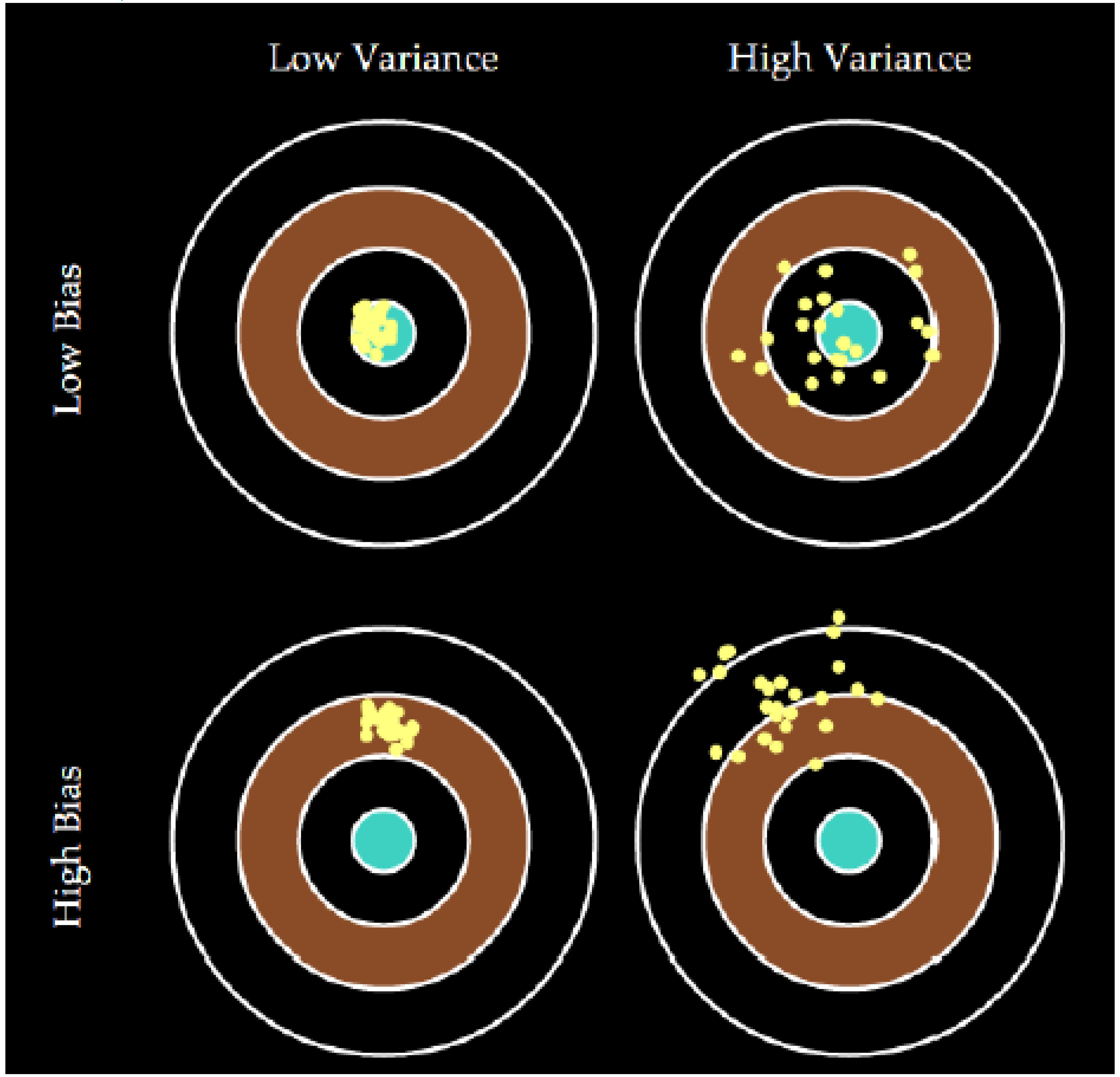
Regression :

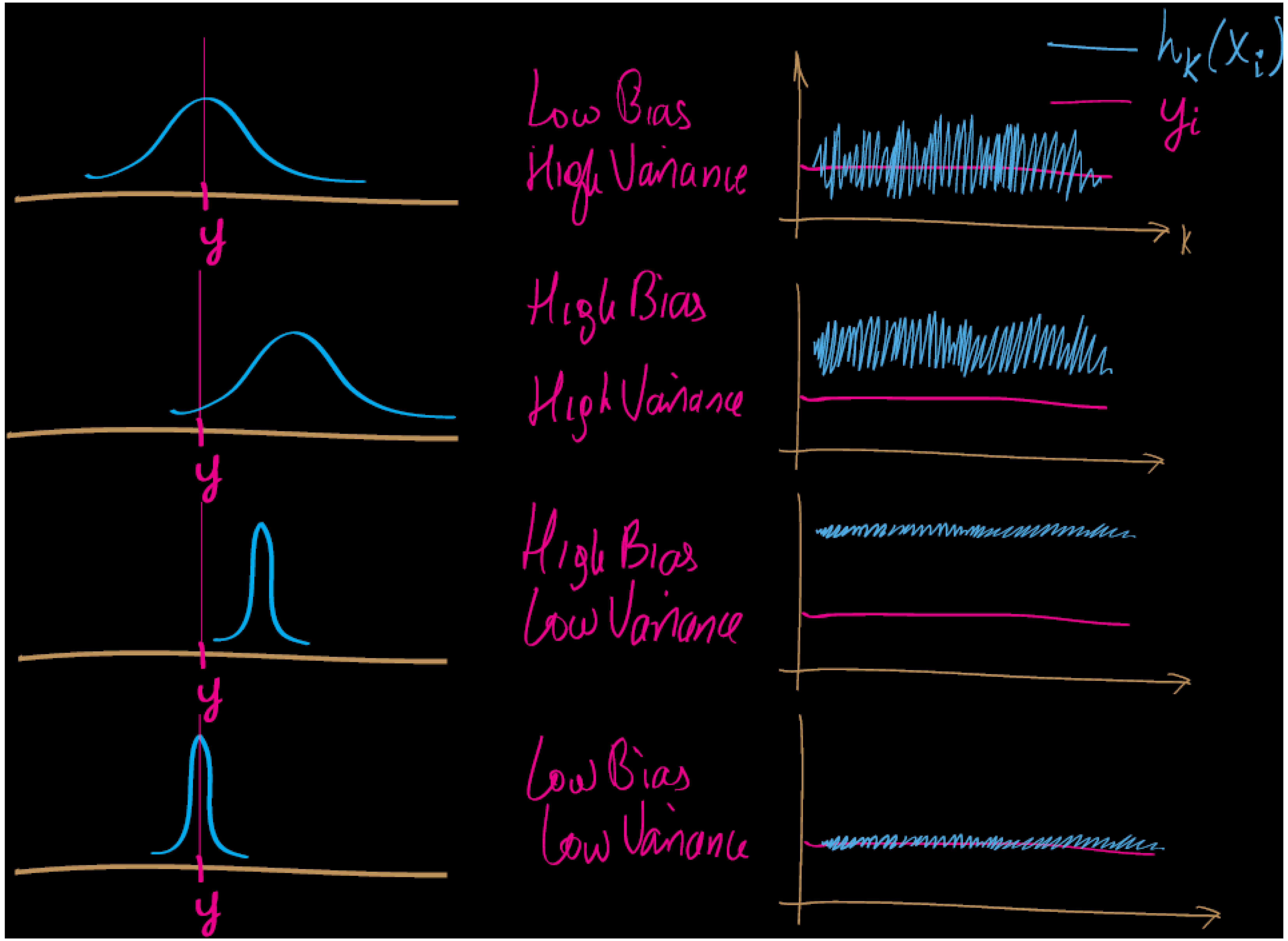
$$\epsilon_{test} = \frac{1}{n}\sum_{i}^{n} (y_i - h(x_i))^2$$

Generalization Error Reformulation

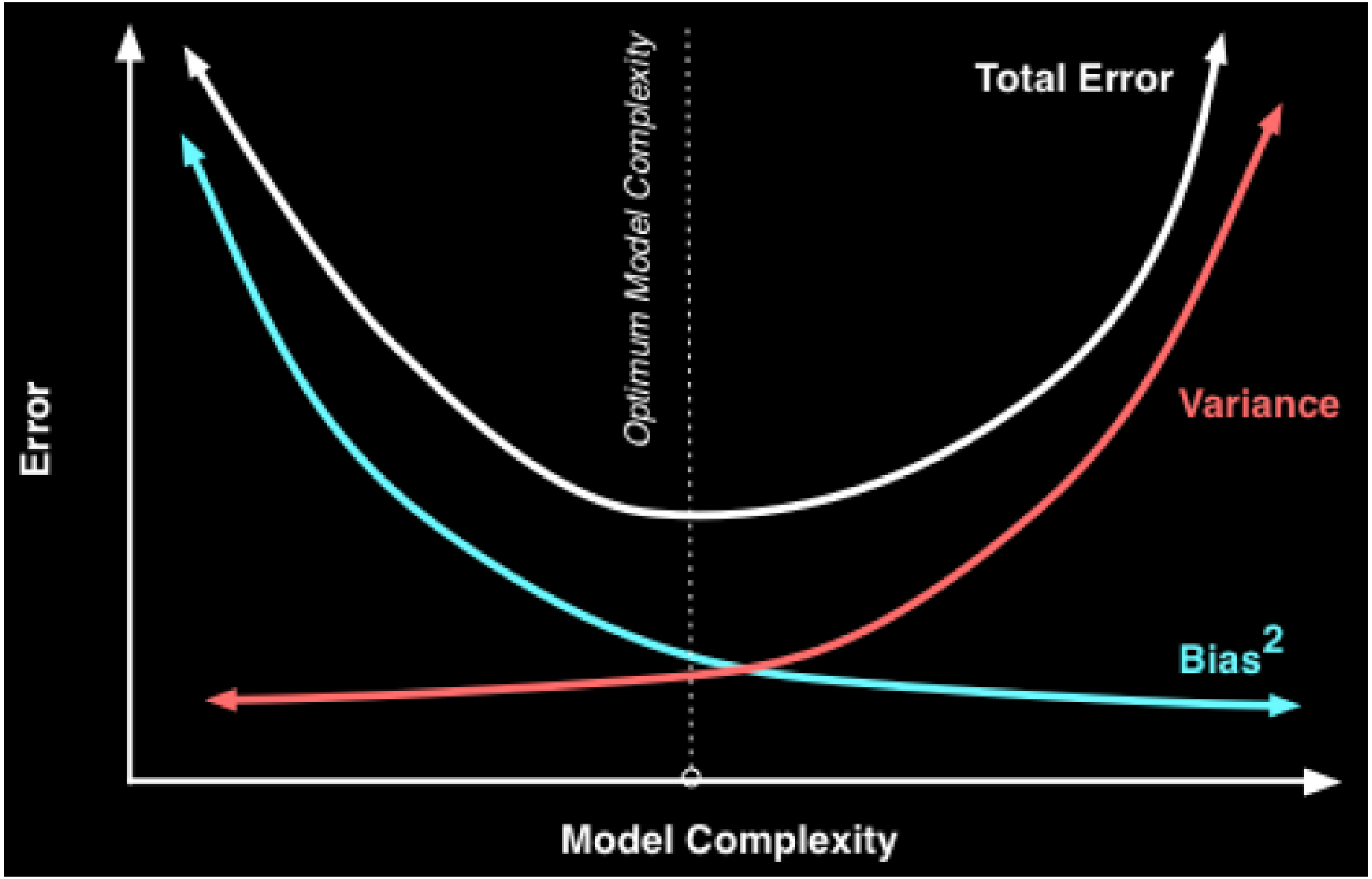
$$
\begin{aligned}
& \bar{\epsilon}_{test}(x_i) \\
= \quad & \frac{1}{T}\sum_{t=1}^{T}(y_i - h_t(x_i))^2 \\
= \quad & \mathbb{E}[(y_i - h_S(x_i))^2] \\
= \quad & \underbrace{(y_i - \mathbb{E}_S[h_S(x_i)])^2}\, bias^2 \\
& + \underbrace{\mathbb{E}_S[(h_s(x_i) - \mathbb{E}_S[h_S(x_i)])^2]}\, variance
\end{aligned}
$$

Bias versus Variance

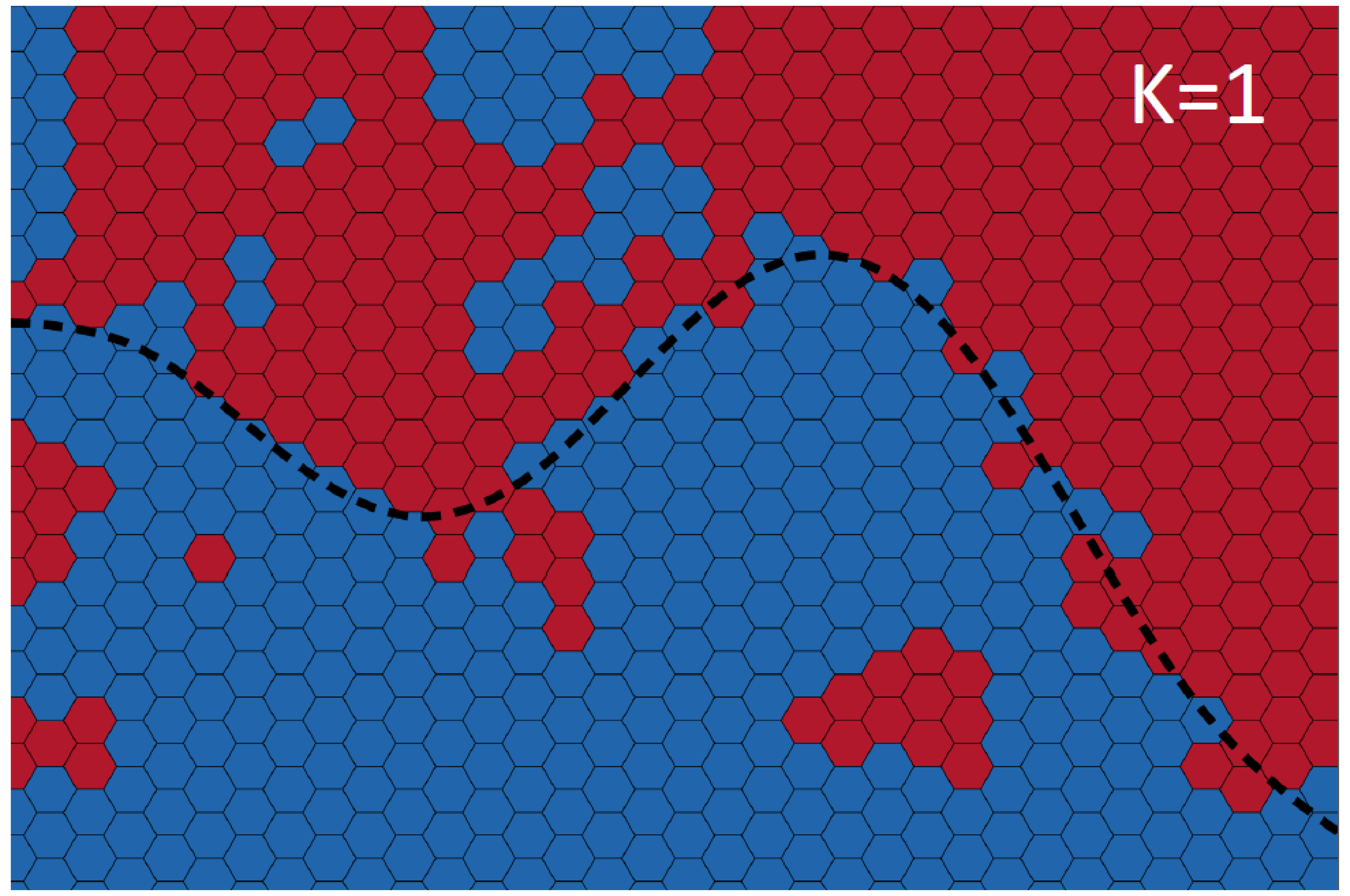


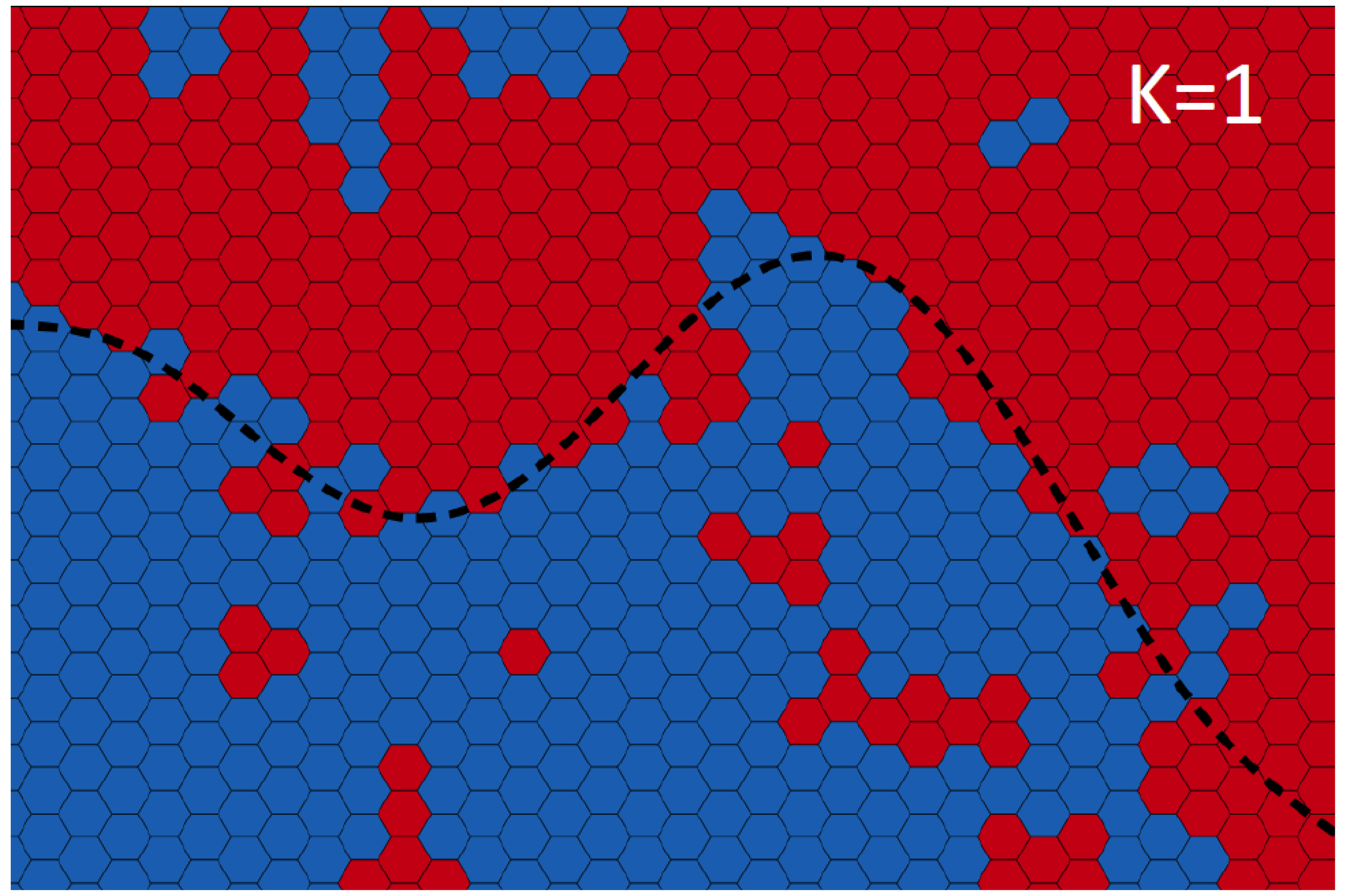


Bias versus Variance



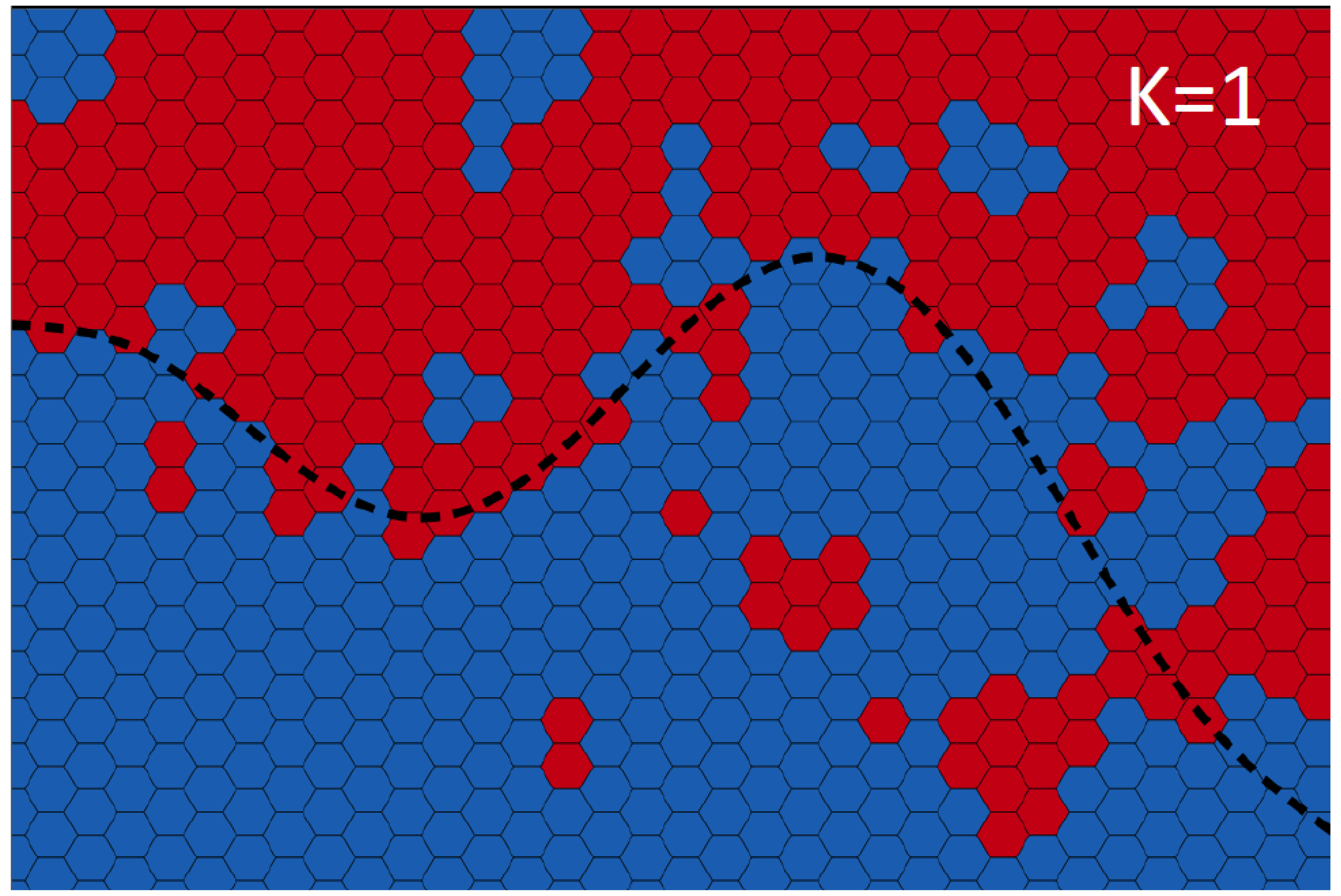
KNN and Model Complexity

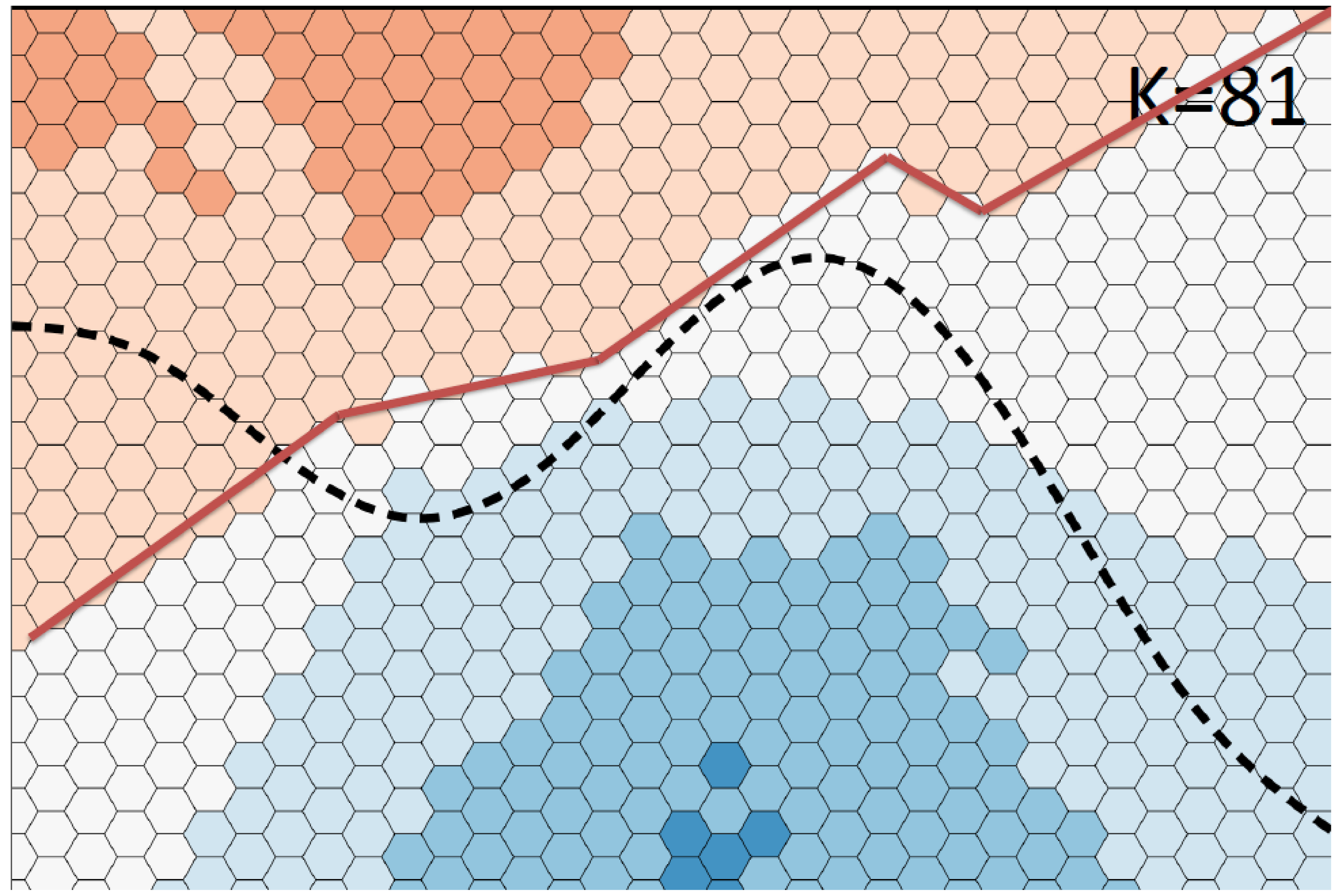


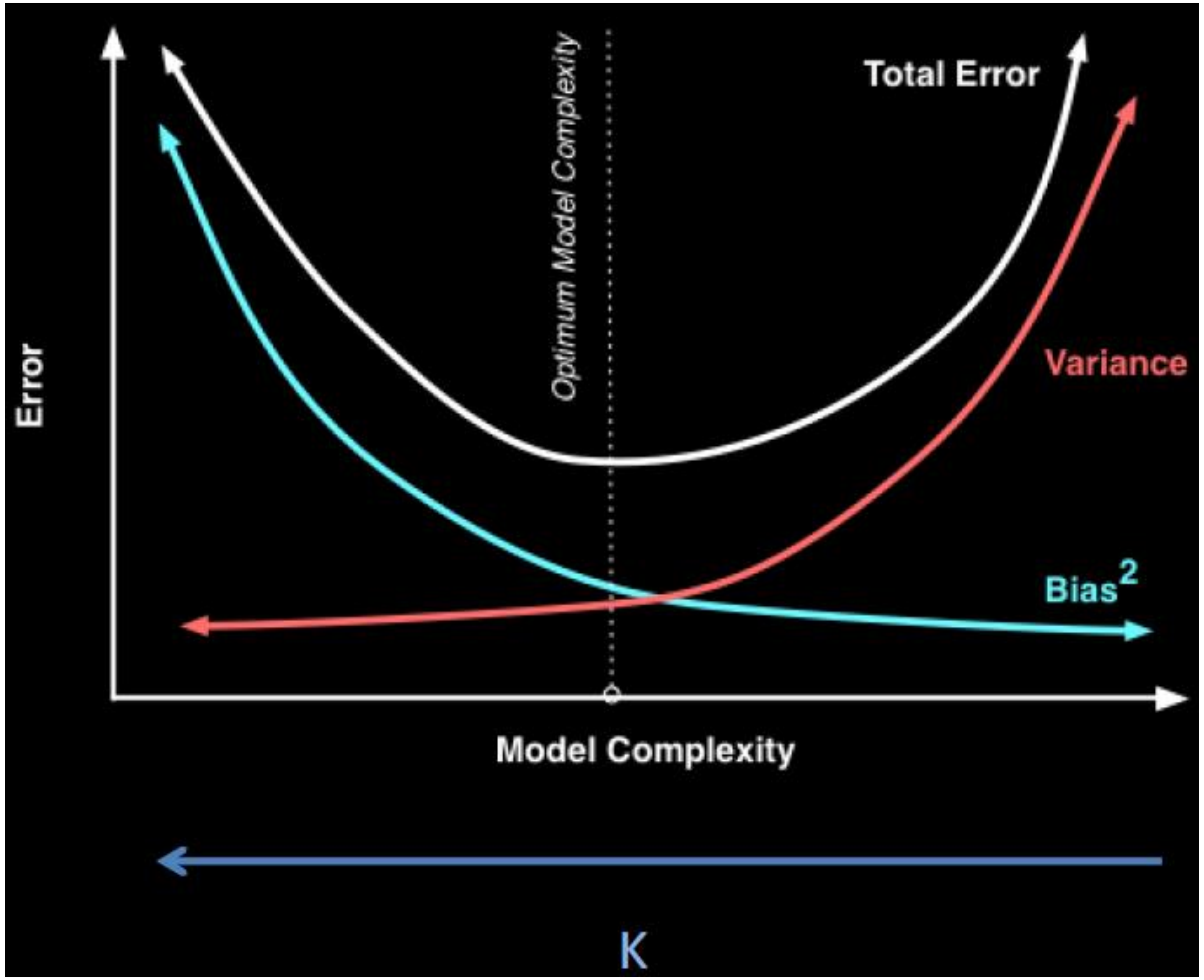


K=1:sensitive to noises;

K=81: stable







Generalization Error with Noise

$$y_i = f(x_i) + \gamma, \gamma \sim \mathcal{N}(0, \sigma^2)$$

$$\begin{aligned}
& \mathbb{E}_S[(y_i - h_S(x_i))^2] \\
= \ & \mathbb{E}_S[(f(x_i) - (h_S(x_i) - \gamma))^2] \\
= \ & (f(x_i) - \mathbb{E}_S[h_S(x_i) - \gamma])^2 + \mathbb{E}_S[((h_S(x_i) - \gamma) - \mathbb{E}_S[h_S(x_i) - \gamma])^2] \\
= \ & (f(x_i) - \mathbb{E}_S[h_S(x_i)])^2 + \mathbb{E}_S[(h_S(x_i) - \mathbb{E}_S[h_S(x_i)] - \gamma)^2] \\
= \ & (f(x_i) - \mathbb{E}[h_S(x_i)])^2 \\
& + \mathbb{E}_S[(h_S(x_i) - \mathbb{E}_S[h_S(x_i)])^2 - 2\gamma(h_S(x_i) - \mathbb{E}_S[h_S(x_i)]) + \gamma^2] \\
= \ & \underbrace{(f(x_i) - \mathbb{E}_S[h_S(x_i)])^2}\, bias^2 \\
& + \underbrace{\mathbb{E}_S[(h_s(x_i) - \mathbb{E}_S[h_S(x_i)])^2]}\, variance \\
& + \underbrace{\mathbb{E}_S[\gamma^2]}\, noise
\end{aligned}$$

Why does Bagging work?

$$h_S(x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x)$$

$$\begin{aligned}
\underbrace{y_i - \mathbb{E}_S[h_S(x_i)]}\, bias \\
&= y_i - \tfrac{1}{T}\textstyle\sum_{t=1}^{T} \mathbb{E}_S[h_t(x_i)] \\
&= \tfrac{1}{T}\textstyle\sum_{t=1}^{T}(y_i - \mathbb{E}_S[h_t(x_i)]) \\
&= \tfrac{1}{T}\textstyle\sum_{t=1}^{T} Bias(h_t, x)
\end{aligned}$$

$$\begin{aligned}
\underbrace{\mathbb{E}_S[(h_S(x_i) - \mathbb{E}_S[h_S(x_i)])^2}\, variance \\
&= \mathbb{E}_S[(\tfrac{1}{T}\textstyle\sum_{t=1}^{T} h_t(x_i) - \mathbb{E}_S[\tfrac{1}{T}\textstyle\sum_{t=1}^{T} h_t(x_i)])^2] \\
&= \mathbb{E}_S[\tfrac{1}{T^2}(\textstyle\sum_{t=1}^{T}(h_t(x_i) - \mathbb{E}_S[h_t(x_i)]))^2] \\
&\approx \tfrac{1}{T}\mathbb{E}_S[(h_t(x_i) - \mathbb{E}_S[h_t(x_i)])^2] \\
&= \tfrac{1}{T}Var(h_t, x_i)
\end{aligned}$$

Bagging has the approximately the same bias, but reduces variance of overall models![6]

The individual models with relatively smaller bias should be selected to maximize the variance-reducing effect of ensemble learning, since the variance can be removed by averaging. If greater weight is given to the models making better predictions, the error can be reduced further.[3]

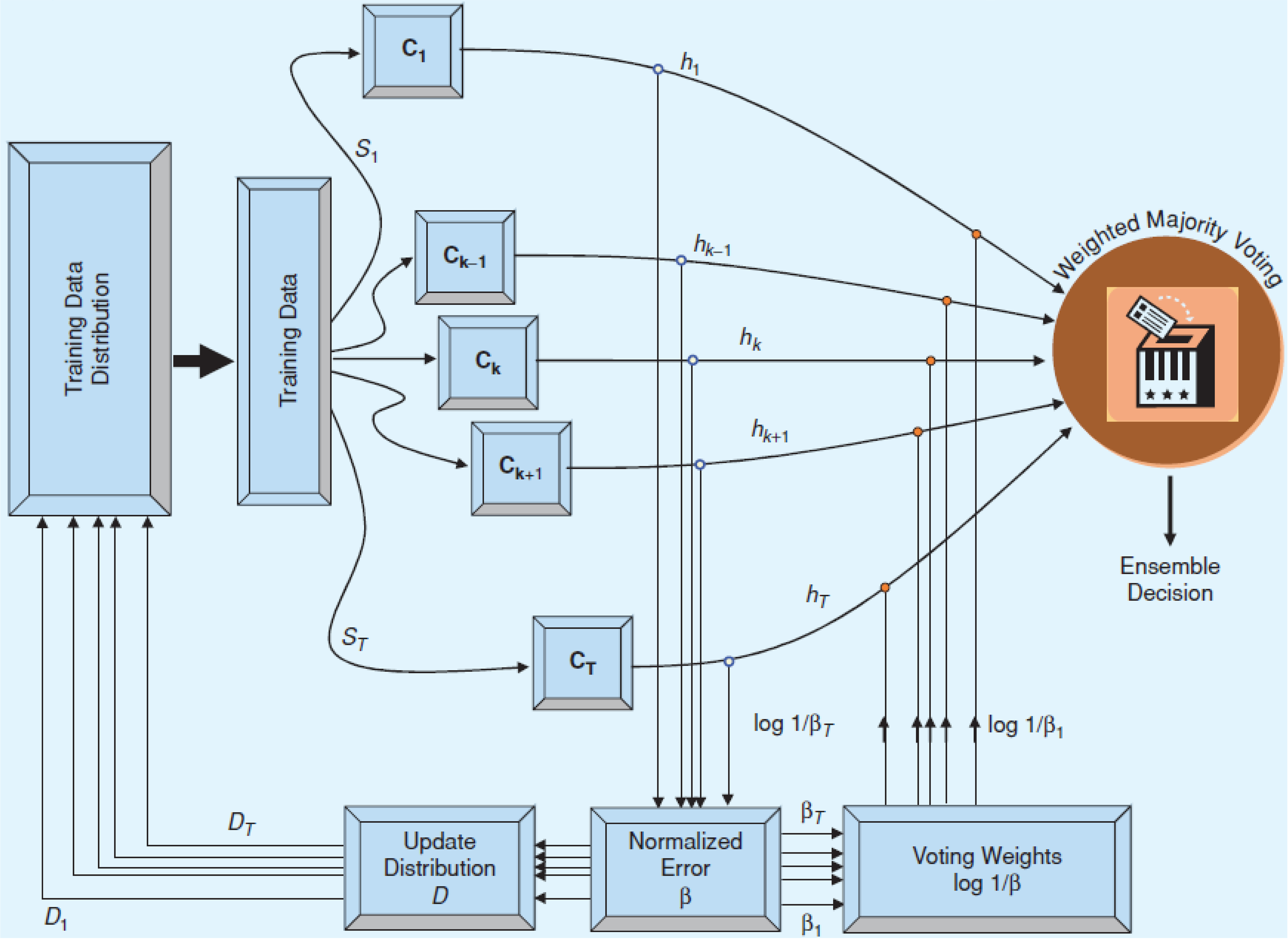
Introduction to AdaBoost

Yunfei WANG
yunfeiwang@hust.edu.cn

[1]School of Computer Science & Technology
Huazhong University of Science & Technology

October 22, 2013

AdaBoost (Adaptive Boosting)



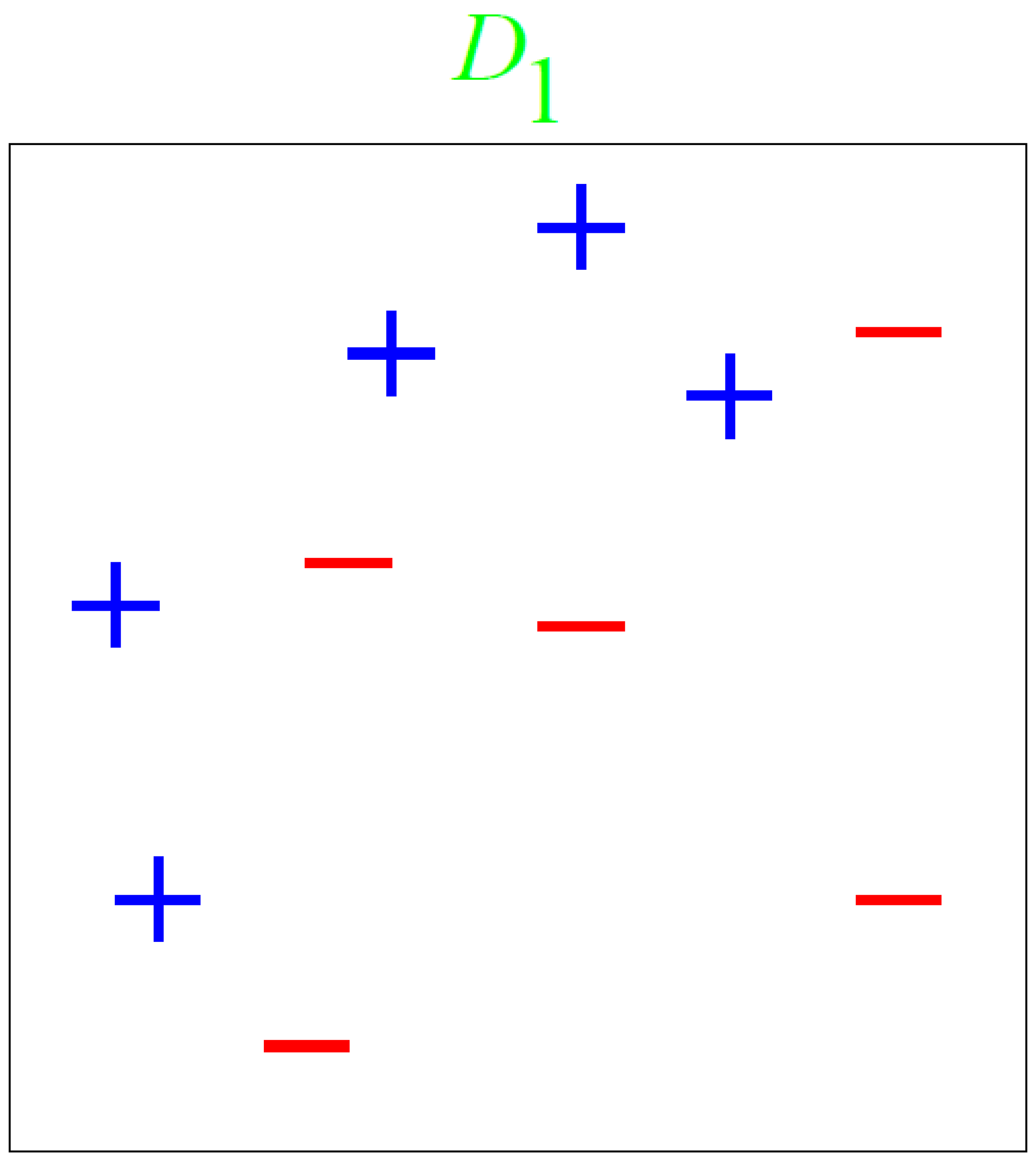
① Update distribution for each training sample: Increase the weight of misclassified samples and decrease that of correctly classified ones.

② Subsequent classifiers focus on misclassified samples via weighted misclassified penalty.

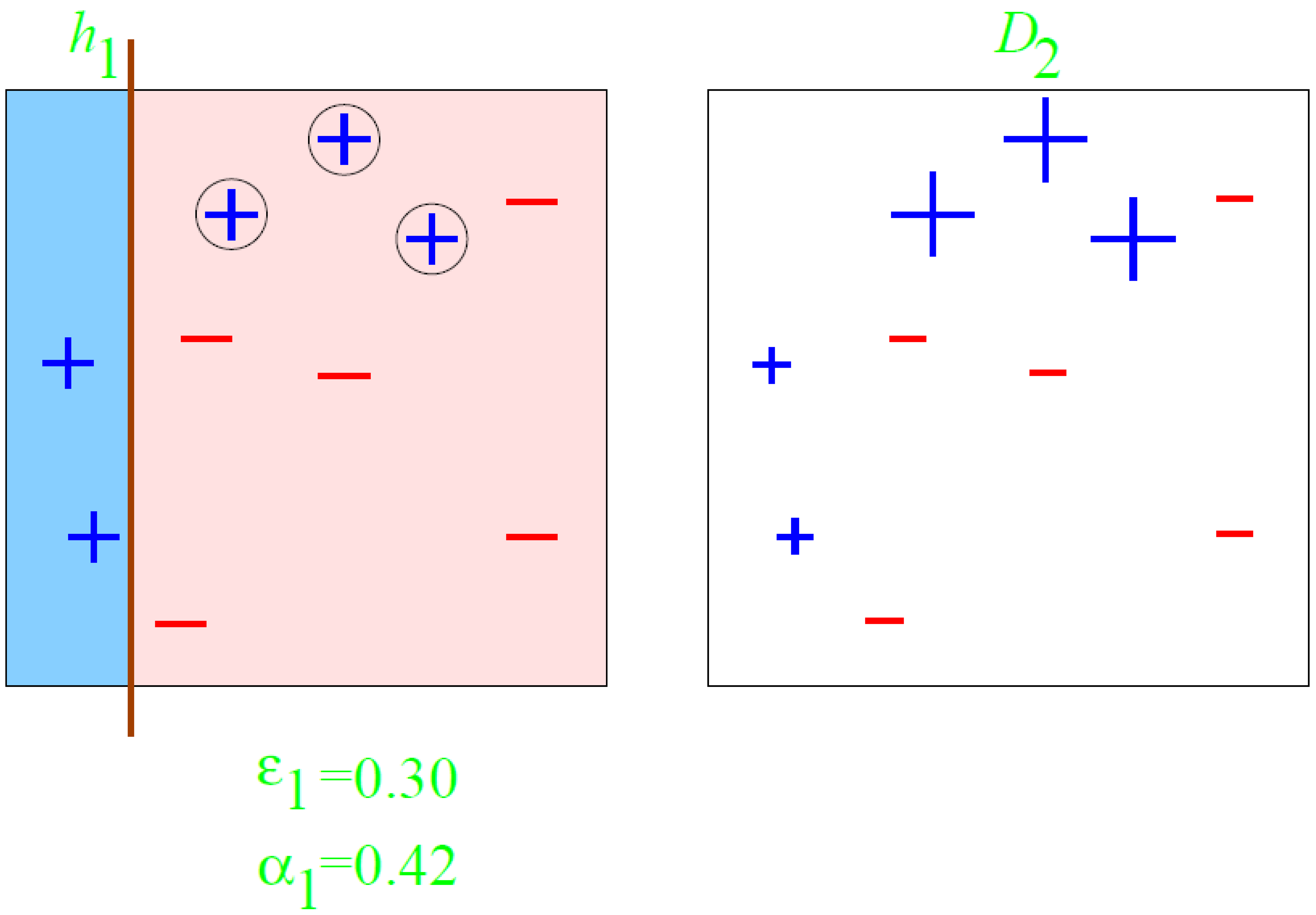
③ Weak classifiers are combined via weighted majority voting.

- Sensitive to noisy data and outliers.
- Less susceptible to the over-fitting problem.

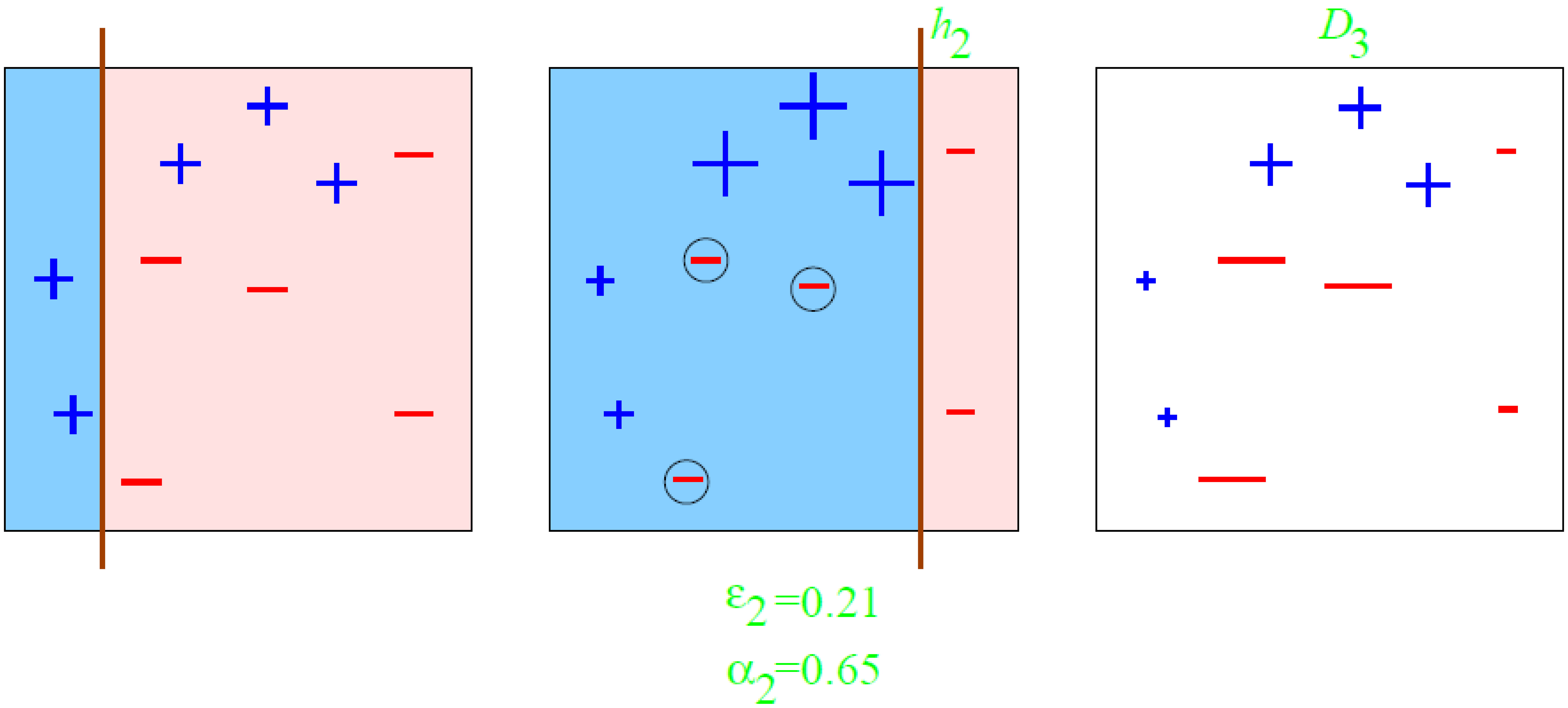
AdaBoost – Toy Example



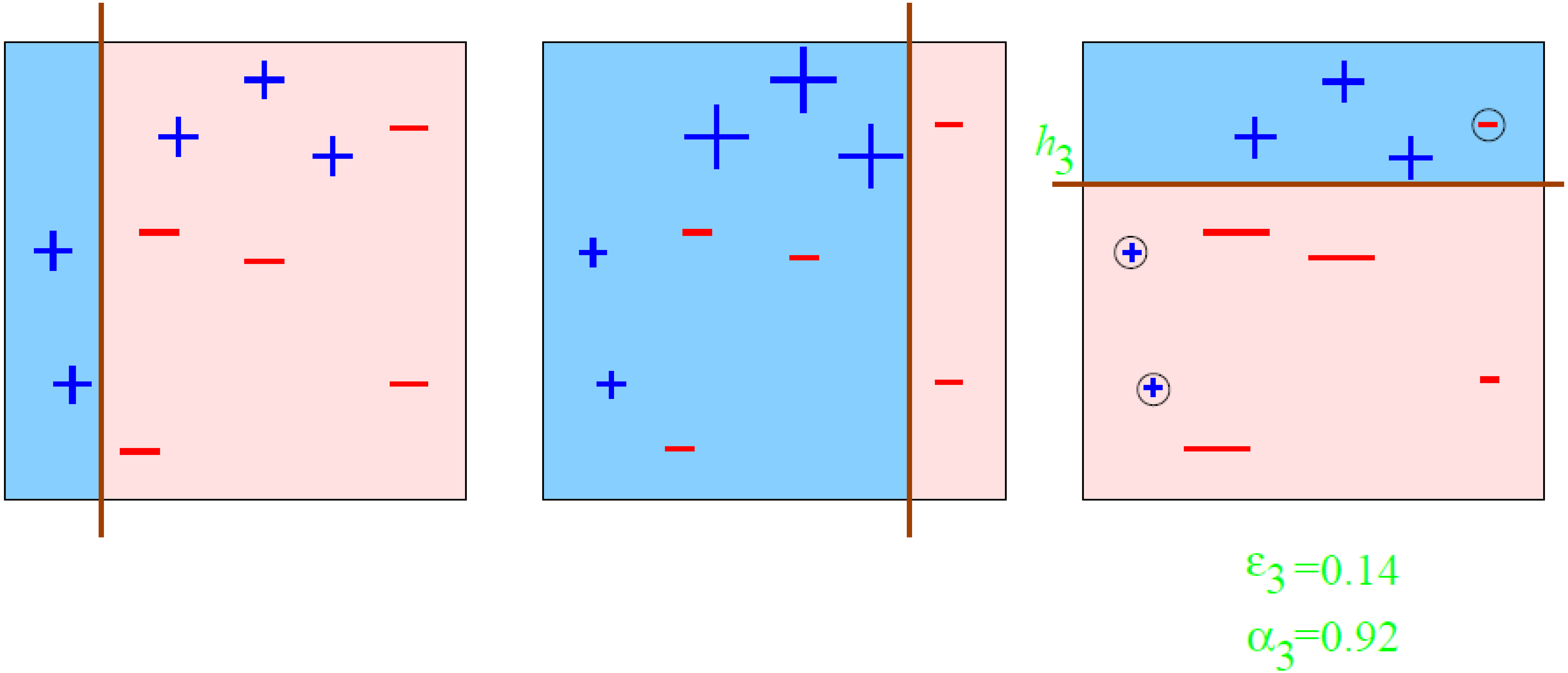
AdaBoost – Toy Example



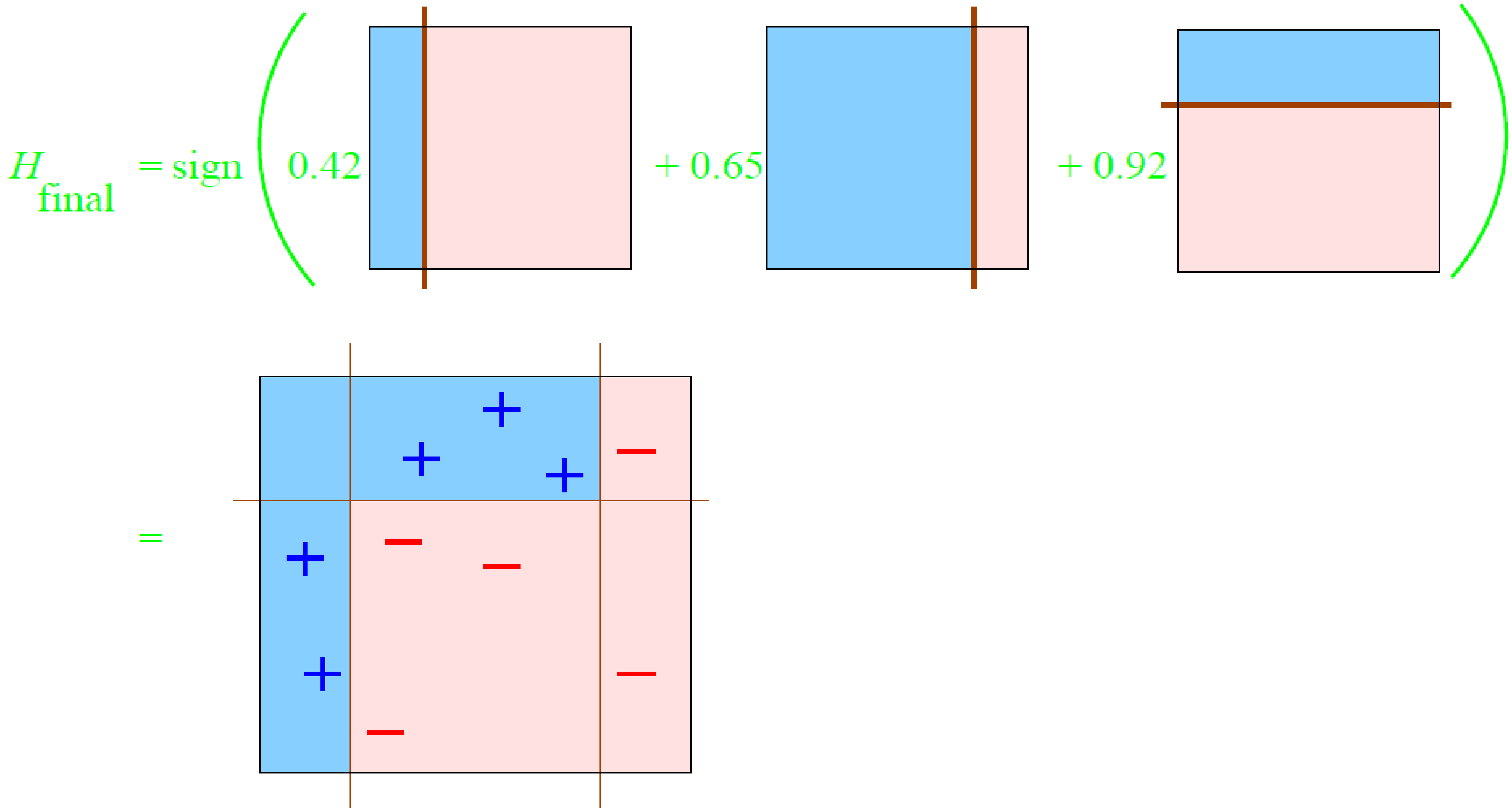
AdaBoost – Toy Example



AdaBoost – Toy Example



AdaBoost – Toy Example



What's AdaBoost

AdaBoost(adaptive boosting) is an algorithm for constructing a "strong" classifier as linear combination of "weak" classifiers $h_m(x) \in \{-1, 1\}$

$$f(x) = \sum_{m=1}^{L} \alpha_m h_m(x) \tag{1}$$

where α_m is the contribution factor of $h_m(x)$.

Final classifier: $H(x) = sign(f(x))$.

AdaBoost is adaptive in the sense that subsequent classifiers are chosen in favor of samples misclassified by previous classifiers.

As long as a classifier is slightly better than random, it can improve the final model. Even classifiers with higher error rate will be help, since they'll have negative coefficients and behave like their inverse.

Scouting is done by testing each classifier with a training set of N samples $\mathcal{S} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N) | y_i \in \{-1, 1\}\}$.

	1	2	3	$\cdots$	L
x_1	0	0	1	$\cdots$	1
x_2	1	0	0	$\cdots$	0
x_3	0	1	1	$\cdots$	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
x_N	1	0	0	$\cdots$	0

Table: Scouting Result of L Classifiers

0-Correct 1-Wrong

Modelling

At the m-th iteration, we have included $m-1$ classifiers, the linear combination of classifiers is

$$f_{m-1}(x) = \alpha_1 h_1(x) + \alpha_2 h_2(x) + \cdots + \alpha_{m-1} h_{m-1}(x) \tag{2}$$

We want to recruit the next classifier

$$f_m(x) = f_{m-1}(x) + \alpha_m h_m(x) \tag{3}$$

The exponential loss of current classifier $sign(f_m(x))$

$$\mathcal{L}_m = \sum_{i=1}^{N} e^{-y_i(f_{m-1}(x_i)+\alpha_m h_m(x_i))} \tag{4}$$

where α_m and h_m are to be determined in an optimal way.

How to choose h_m?

According to the results of classifier h_m, we split $\mathcal{L}_m$ into two parts

$$\mathcal{L}_m = \sum_{i:y_i=h_m(x_i)} w_i^{(m)} e^{-\alpha_m} + \sum_{i:y_i\neq h_m(x_i)} w_i^{(m)} e^{\alpha_m} = W_c e^{-\alpha_m} + W_e e^{\alpha_m} \tag{5}$$

where $w_i^{(m)} = e^{-y_i f_{m-1}(x_i)}$ is the weight assigned to each sample.

When selecting h_m, we change the form of $\mathcal{L}_m$

$$\mathcal{L}_m = (W_c + W_e)e^{-\alpha_m} + W_e(e^{\alpha_m} - e^{-\alpha_m}) \tag{6}$$

$W = W_c + W_e$ is the sum of weights of all samples, which is fixed in current iteration. We'll choose the classifier with lowest $W_e(\alpha_m > 0)$ or highest $W_e(\alpha_m < 0)$.

What is the optimal α_m for h_m?

Having picked the m-th classifier, we need to determine its weight α_m

$$\frac{\partial \mathcal{L}_m}{\partial \alpha_m} = -W_c e^{-\alpha_m} + W_e e^{\alpha_m} \tag{7}$$

Setting the partial derivative to zero and rescaling it by e^{α_m}

$$-W_c + W_e e^{2\alpha_m} = 0 \tag{8}$$

The optimal α_m is thus

$$\alpha_m = \frac{1}{2} log(\frac{W_c}{W_e}) = \frac{1}{2} log(\frac{W - W_e}{W_e}) = \frac{1}{2} log(\frac{1 - \epsilon_m}{\epsilon_m}) \tag{9}$$

where $\epsilon_m = W_c/W_e$ is the rate of error given the weights of all samples.

Reweighting

Reweighting formula

$$w_i^{(m+1)} = \frac{w_i^{(m)} e^{-y_i \alpha_m h_m(x_i)}}{\mathcal{Z}_m} = \frac{e^{-y_i \sum_{q=1}^{m-1} \alpha_q h_q(x_i)} \cdot e^{-y_i \alpha_m h_m(x_i)}}{N \prod_{q=1}^{m} \mathcal{Z}_q} \tag{10}$$

Effect on training set

Increase the weight of wrongly classified examples by e^{α_m};
Decrease the weight of correctly classified examples by $e^{-\alpha_m}$;

Effect on current classifier h_m

Weighted error of h_m in next iteration is $1/2$. Proof:

$$\frac{\sum_{i:y_i \neq h_m(x_i)} w_i^{(m+1)}}{\sum_{i:y_i = h_m(x_i)} w_i^{(m+1)}} = \frac{\sum_{i:y_i \neq h_m(x_i)} e^{\alpha_m}}{\sum_{i:y_i = h_m(x_i)} e^{-\alpha_m}} = \frac{e^{2\alpha_m} \epsilon_m}{1 - \epsilon_m} = 1 \tag{11}$$

Algorithm of AdaBoost

Input: Training set $\mathcal{S} = \{(x_1, y_1), \cdots, (x_N, y_N) | x_i \in \mathbf{X}, y_i \in \{-1, 1\}\}$
Input: Number of iterations T, threshold β;

Initialize weights $w_i^{(1)} = 1/N$ for $i = 1, 2, \cdots, N$;
for $m = 1, 2, \cdots, T$ **do** `// ` $\mathcal{H}$ ` is the family of weak classifiers`
  $h_m = \underset{h_m \in \mathcal{H}}{arg \max} |0.5 - \epsilon_m|$,where $\epsilon_m = \sum_{i=1}^{N} w_i^t I(y_i \neq h_t(x_i))$;
  If $|0.5 - \epsilon_m| \leq \beta$ break;
  Choose $\alpha_m \in \mathbb{R}$,typically $\alpha_m = \frac{1}{2} log(\frac{1-\epsilon_m}{\epsilon_m})$;
  Update weights for each sample $w_i^{(m+1)} = \frac{w_i^{(m)} e^{-y_i \alpha_m h_m(x_i)}}{\mathcal{Z}_m}$,where $\mathcal{Z}_m$ is the normalization factor;
end
return the final strong classifier:

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x)) \tag{12}$$

Analysing Training Error I

Normalization constant at t-th iteration

$$
\begin{aligned}
\mathcal{Z}_q &= \sum_{i=1}^{N} w_i^{(q)} \exp(-y_i \alpha_i h_m(x_i)) \\
&= \sum_{i: y_i \neq h_m(x_i)} w_i^{(q)} \exp(-y_i \alpha_i h_m(x_i)) \\
&\quad + \sum_{i: y_i = h_m(x_i)} w_i^{(q)} \exp(-y_i \alpha_i h_m(x_i)) \\
&= \epsilon_q \exp(-\alpha_q) + (1-\epsilon_q)\exp(\alpha_q) \\
&= 2\sqrt{\epsilon_q(1-\epsilon_q)} \\
&= \sqrt{1-4\gamma_q^2} \leq \exp(-2\gamma_q^2)
\end{aligned}
\tag{13}
$$

where $\gamma_q = 1/2 - \epsilon_q \in [-1/2, 1/2]$ measures how much better than random is h_t's performance.

Weight for each sample after including m-th classifier into consideration

$$
w_i^{(m+1)} = \frac{\exp(-y_i \sum_{q=1}^{m} \alpha_q h_q(x_i))}{N \prod_{q=1}^{m} \mathcal{Z}_q} = \frac{\exp(-y_i f_m(x_i))}{N \prod_{q=1}^{m} \mathcal{Z}_q} \tag{14}
$$

$$h_m(x_i) \neq y_i \Rightarrow y_i f_m(x_i) < 0 \Rightarrow \exp(-y_i f_m(x_i)) > 1 \tag{15}$$

$$h_m(x_i) = y_i \Rightarrow y_i f_m(x_i) > 0 \Rightarrow 0 < \exp(-y_i f_m(x_i)) < 1 \tag{16}$$

Upper bound of training error

$$\begin{aligned} TrainingError &= \tfrac{1}{N}\textstyle\sum_{i=1}^{N} I\{y_i \neq h_m(x_i)\} \\ &\leq \tfrac{1}{N}\textstyle\sum_{i=1}^{N} \exp(-y_i f_m(x_i)) \\ &= \textstyle\sum_{i=1}^{N} w_i^{(m+1)} \prod_{q=1}^{m} \mathcal{Z}_q \\ &= \textstyle\prod_{q=1}^{m} \mathcal{Z}_q \\ &= \textstyle\prod_{q=1}^{m} \sqrt{1-4\gamma_q^2} \\ &\leq \exp(-2\textstyle\sum_{q=1}^{m} \gamma_q^2) \end{aligned} \tag{17}$$

Therefore,if each as long as each weak classifier is slightly better or worse than random $\gamma_q > 0$,then training error drops fast.

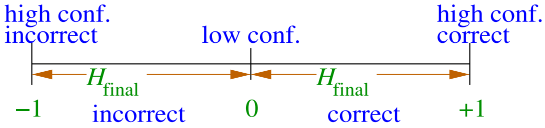
A Better Story: The Margins Explanation

[with Freund, Bartlett & Lee]

- key idea:
  - training error only measures whether classifications are right or wrong
  - should also consider confidence of classifications
- recall: H_{final} is weighted majority vote of weak classifiers
- measure confidence by margin = strength of the vote

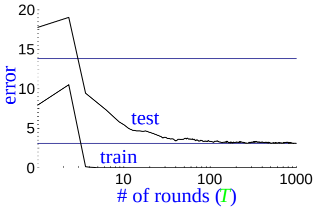
  = (fraction voting correctly) − (fraction voting incorrectly)

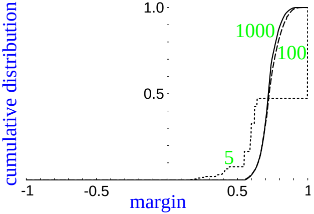
$$\frac{y \sum_t \alpha_t h_t(x)}{\sum_t \alpha_t}$$



Empirical Evidence: The Margin Distribution

- margin distribution
  = cumulative distribution of margins of training examples





	# rounds		
	5	100	1000
train error	0.0	0.0	0.0
test error	8.4	3.3	3.1
% margins ≤ 0.5	7.7	0.0	0.0
minimum margin	0.14	0.52	0.55

More Technically...

- with high probability, $\forall\theta > 0$:

$$\text{generalization error} \leq \hat{\Pr}[\text{margin} \leq \theta] + \tilde{O}\left(\frac{\sqrt{d/m}}{\theta}\right)$$

($\hat{\Pr}[\,] =$ empirical probability)

  - bound depends on
    - $m = \#$ training examples
    - $d =$ "complexity" of weak classifiers
    - entire distribution of margins of training examples
- $\hat{\Pr}[\text{margin} \leq \theta] \rightarrow 0$ exponentially fast (in T) if (error of h_t on D_t) $< 1/2 - \theta$ ($\forall t$)
  - so: if weak learning assumption holds, then all examples will quickly have "large" margins

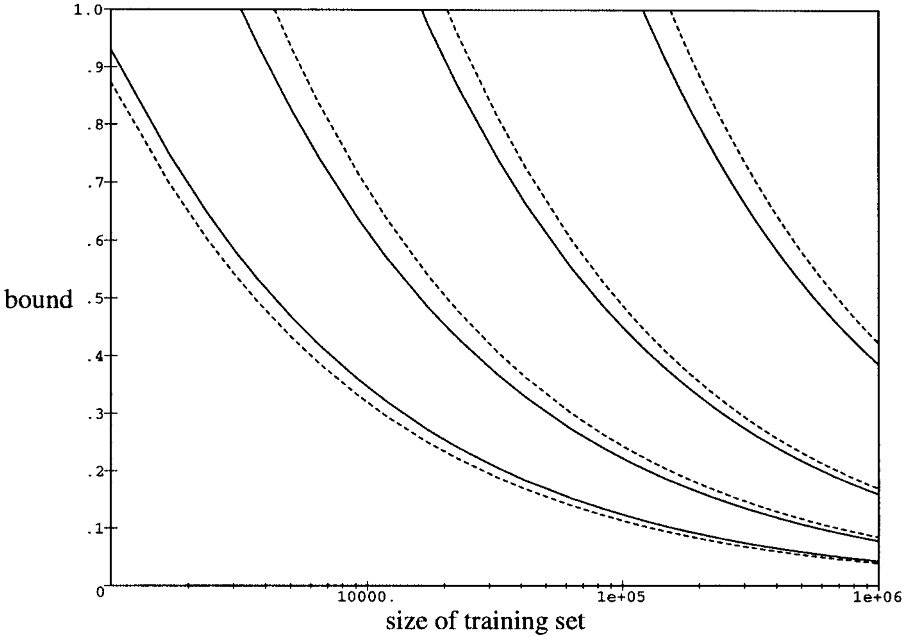


FIG. 3. *A few plots of the second and third terms in the bound given in* (8) (*solid lines*) *and their approximation by the second term in* (9) (*dotted lines*). *The horizontal axis denotes the number of training examples* (*with a logarithmic scale*) *and the vertical axis denotes the value of the bound. All plots are for* $\delta = 0.01$ *and* $|\mathscr{H}| = 10^6$. *Each pair of close lines corresponds to a different value of* θ; *counting the pairs from the upper right to the lower left, the values of* θ *are* $1/20$, $1/8$, $1/4$ *and* $1/2$.

Too Much or Too Little Data

Large volumes of data

1. Partition the data into small subsets;
2. Training different classifiers with different partitions of data;
3. Combining their outputs with an appropriate combination rule

Too little data

1. Drawing overlapping random subsets of the available data
2. Training a different classifier on different bootstrap samples
3. Creating an ensemble

Data Fusion

Data from different sources may provide complementary information. Heterogeneous features cannot be used all together to train a single model.

A suitable combination of such information can lead to improved accuracy of decision compared to the one based on individual data sources.

Ensemble Learning for Missing Features Problem

Percentage of features(15%-40%) used depend on redundancy of features and complexity of task(Larger feature subspace ->Stronger but fewer individual models).[2]

References

[1] Polikar, Robi. "Ensemble based systems in decision making." *Circuits and Systems Magazine, IEEE* 6.3 (2006): 21-45.

[2]Krause, Stefan, and Robi Polikar. "An ensemble of classifiers approach for the missing feature problem." *Neural Networks, 2003*.

[3] Sewell, Martin. "Ensemble learning." *RN* 11.02 (2008). *Proceedings of the International Joint Conference on*. Vol. 1. IEEE, 2003.

[4] http://lovebingkuai.diandian.com/post/2013-04-28/40050551522

[5] Dietterich, Thomas G. "Ensemble methods in machine learning." *Multiple classifier systems*. Springer Berlin Heidelberg, 2000. 1-15.

[6] Meir, Ron, and Gunnar Rätsch. "An introduction to boosting and leveraging."*Advanced lectures on machine learning*. Springer Berlin Heidelberg, 2003. 118-183.

[7] 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.