Machine Learning

Lecture 8.
Ensemble Methods: Bagging, Boosting

Alireza Rezvanian

Fall 2022

Last update: Nov. 10, 2022

Amirkabir University of Technology (Tehran Polytechnic)

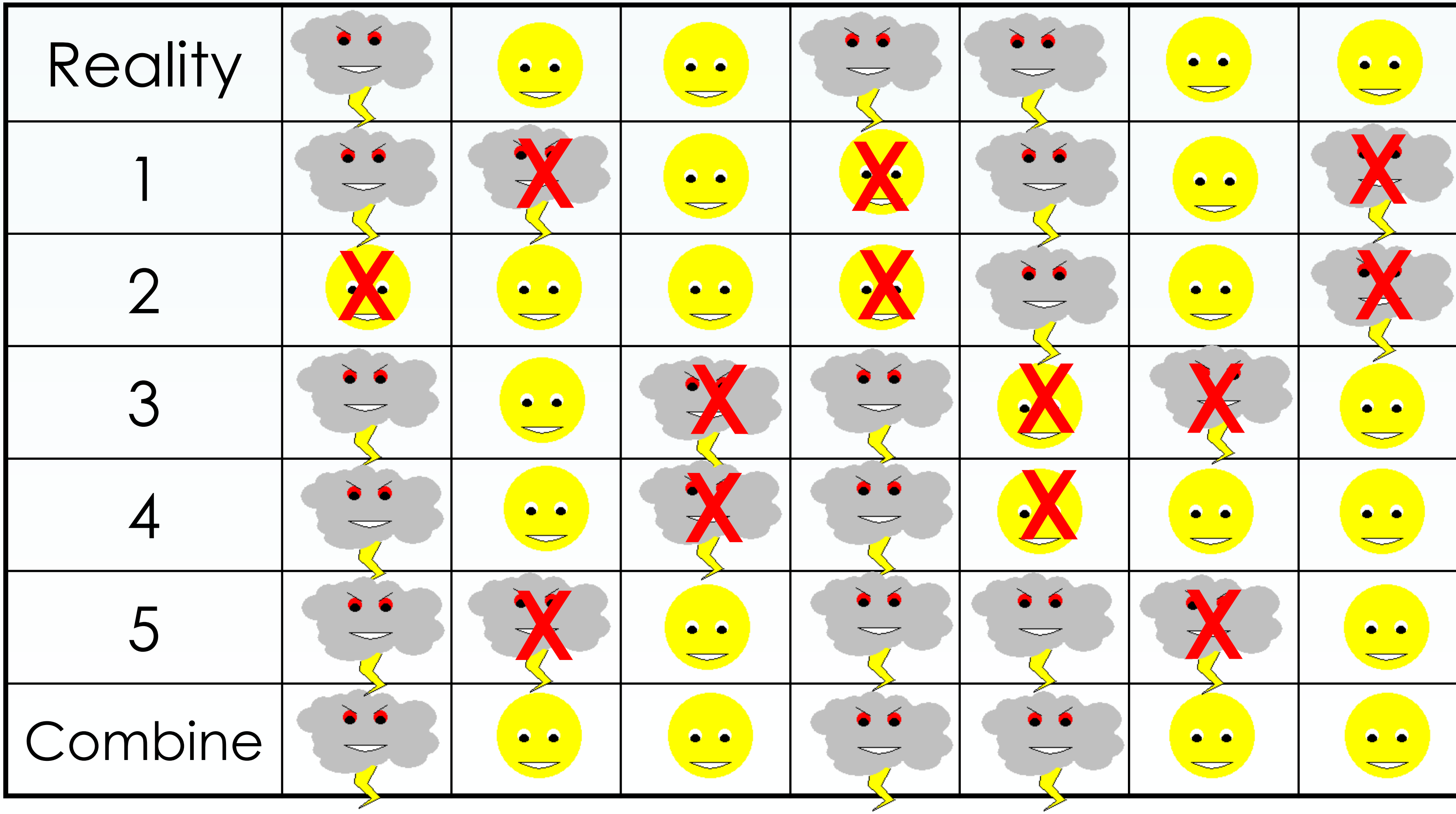
Outline

- Combining multiple learners
- Types of Committee Machines
- Ensemble Averaging: Voting
- Ensemble Methods
  - Bagging
  - Boosting
  - AdaBoost
- Dynamic Methods
  - Mixture of Experts
  - Stacking

Rationale

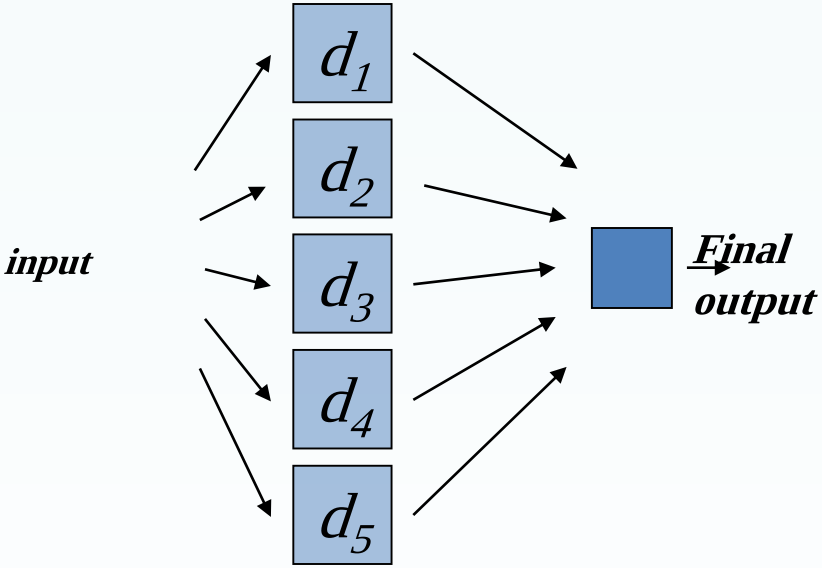
- No Free Lunch Theorem: There is no algorithm that is always the most accurate
  - No learner can succeed on all learnable task – every learner has tasks on which it fails while other learner succeed.
- **Idea**: Generate a group of base-learners which when combined has higher accuracy
- Different learners use different
  - Algorithms: making different hypothesizes
  - Hyperparameters: E.g., different number of hidden neurons in ANN, k in k-NN
  - Representations /Modalities/Views: different features for each learner, multiple sources of information
  - Training sets: variations in datasets or Subproblems
- Diversity vs accuracy

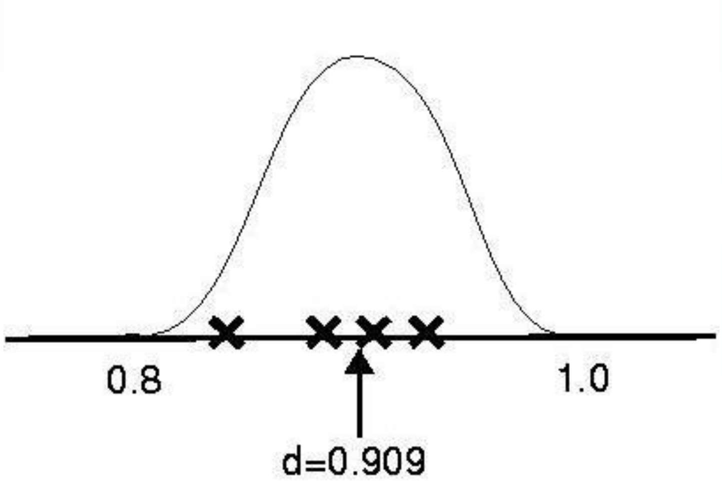
Example: Weather Forecast



Combining multiple learners

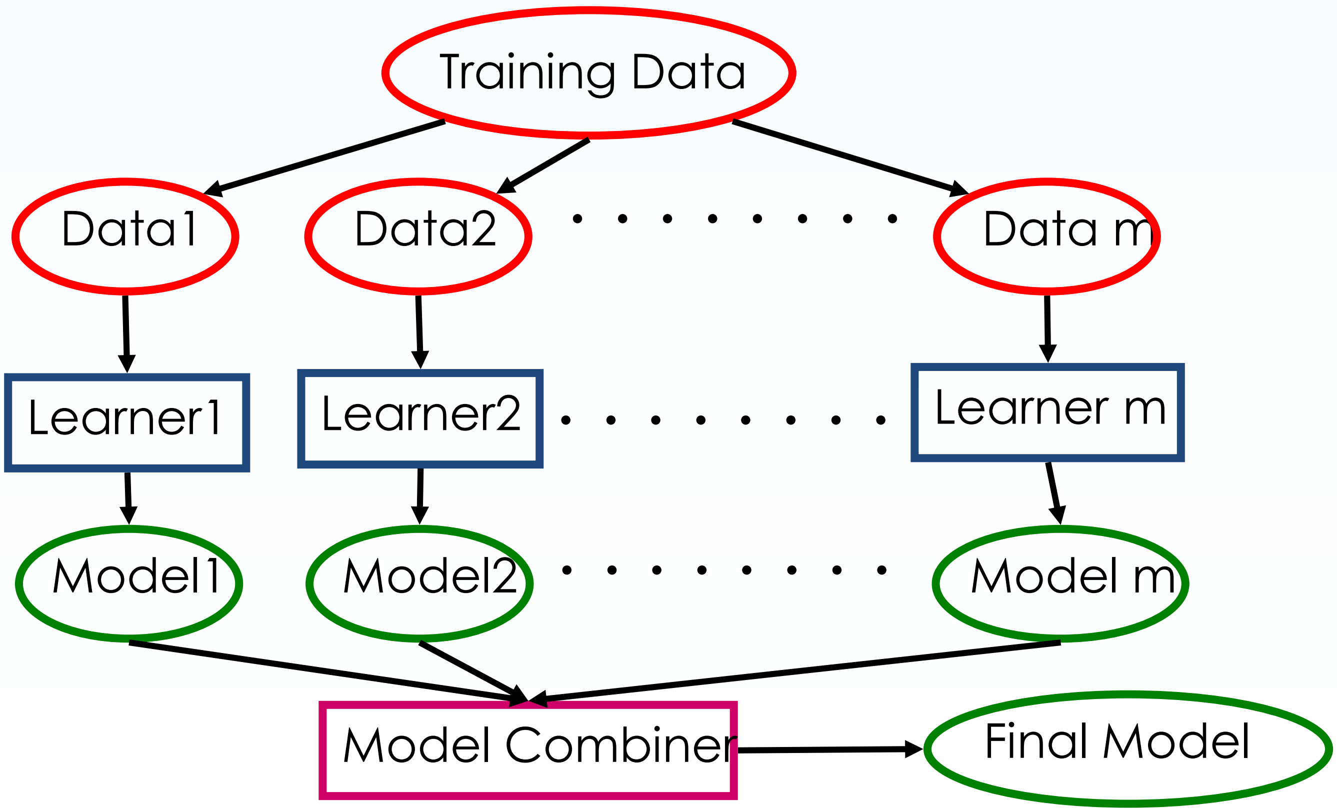
- Lots of different combination methods:
  - Most popular are averaging and majority voting
- Intuitively, it seems as though it should work.
  - We have parliaments of people who vote, and that works …
  - We average guesses of a quantity, and we'll probably be closer…





We want the base learners to be

- Complementary
  - what if they are all the same or very similar
- Reasonably accurate
  - but not necessarily very accurate



Types of Committee Machines

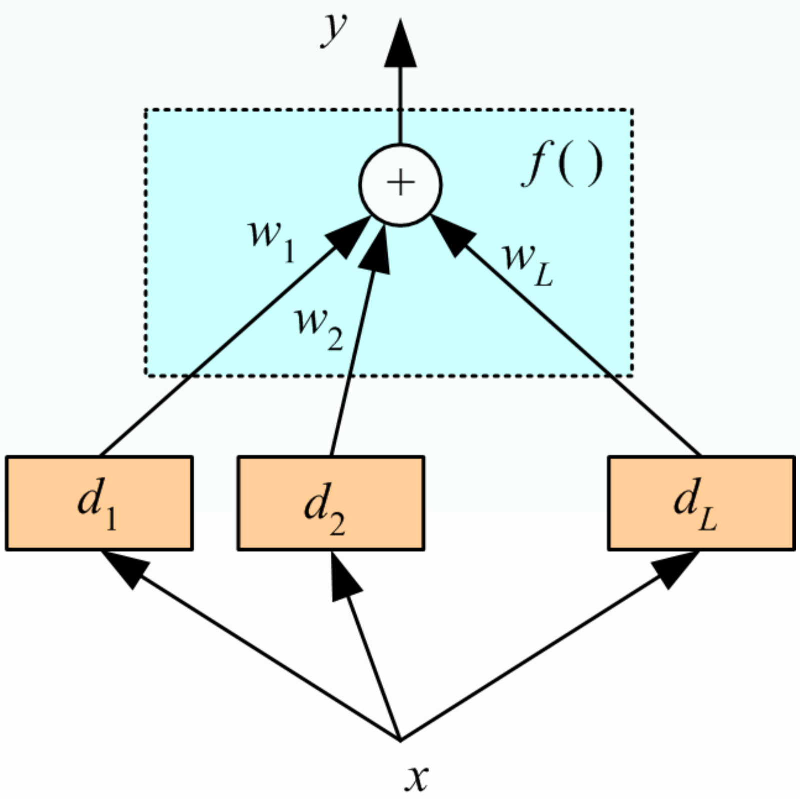
- **Static Structures:** The outputs of several expert (constituent) are combined by a mechanism that does not involve the input signal.
  - **Ensemble averaging**: the expert outputs are linearly combined.
  - **Boosting**: weak learners are combined to give a strong learner.
- **Dynamic Structures:** The input signal is directly involved in actuating the mechanism that integrates/combines the expert outputs.
  - **Mixtures of experts**: the expert outputs are non-linearly combined by some form of **gating system** (which may itself be a neural network)
  - Hierarchical mixture of experts

Ensemble Averaging: Voting

- **Regression**

$$y = \sum_{j=1}^{L} w_j d_j$$

$$w_j \geq 0 \text{ and } \sum_{j=1}^{L} w_j = 1$$



- **Classification**

$$y_i = \sum_{j=1}^{L} w_j d_{ji}$$

- **w_j=1/L**
  - plurality voting
  - majority voting
- **w_j proportional to error rate of classifier:**
  - Learned over a validation set

Ensemble methods

- Product rule
  - Assumes that representations used by different classifiers are conditionally independent
- **Sum rule (voting with uniform weights)**
  - Further assumes that posteriors of class probabilities are close to the class priors
  - Very successful in experiments, despite very strong assumptions
    - Committee machine less sensitive to individual errors
- Min rule
  - Can be derived as an approximation to the product/sum rule
- Max rule

The respective assumptions of these rules are analyzed in Kittler et al. 1998.
The sum-rule is thought to be the best at the moment

Fixed Combination Rules

Rule	Fusion function $f(\cdot)$
Sum	$y_i = \frac{1}{L}\sum_{j=1}^{L} d_{ji}$
Weighted sum	$y_i = \sum_j w_j d_{ji}, w_j \geq 0, \sum_j w_j = 1$
Median	$y_i = \text{median}_j d_{ji}$
Minimum	$y_i = \min_j d_{ji}$
Maximum	$y_i = \max_j d_{ji}$
Product	$y_i = \prod_j d_{ji}$

	C_1	C_2	C_3
d_1	0.2	0.5	0.3
d_2	0.0	0.6	0.4
d_3	0.4	0.4	0.2
Sum	0.2	**0.5**	0.3
Median	0.2	**0.5**	0.4
Minimum	0.0	**0.4**	0.2
Maximum	0.4	**0.6**	0.4
Product	0.0	**0.12**	0.032

Example

- Combining the outputs of 10 networks, the ensemble average achieves an expected error (ε_D) less than the expected value of the average error of the individual networks, over many trials with different data sets
  - 80.3% versus 79.4% (average)
  - 1% diff.

TABLE 7.1 Classification Performances of Individual Experts U in a Committee Machine

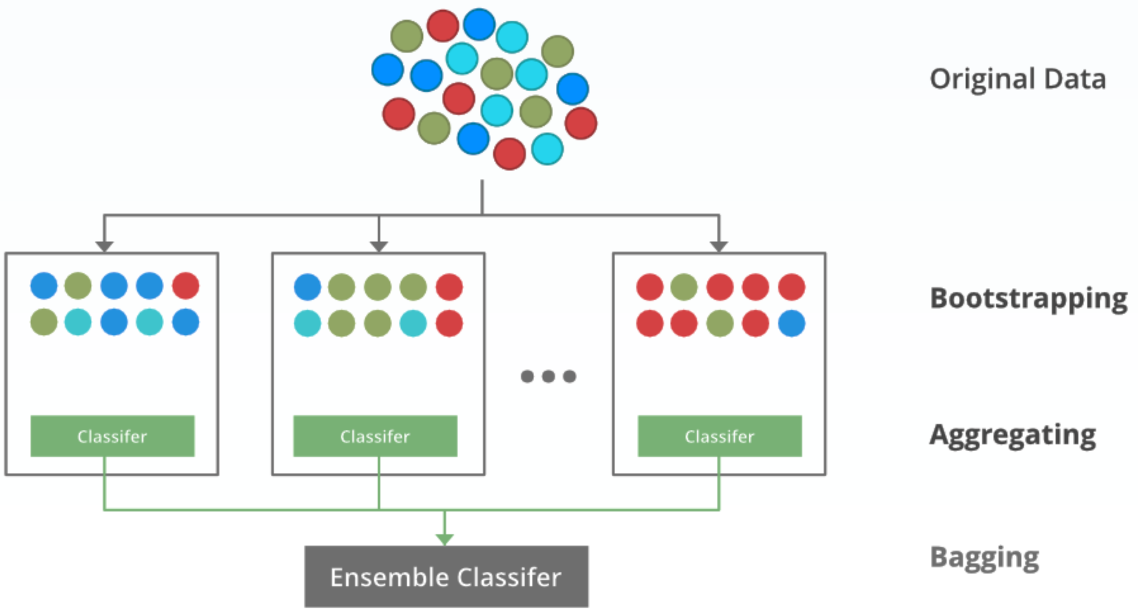
Expert	Correct classificat percentage	
Net 1	80.65	
Net 2	76.91	
Net 3	80.06	
Net 4	80.47	
Net 5	80.44	Avg. 79.4
Net 6	76.89	
Net 7	80.55	
Net 8	80.47	
Net 9	76.91	
Net 10	80.38	

Ensemble Methods: Bagging

- Voting method where base-learners are made different by training over slightly different training sets

Bagging (Bootstrap Aggregating) - Breiman, 1996
    take a training set D, of size N
    for each network / tree / k-nn / etc…
        - build a new training set by sampling N examples,
            randomly with replacement, from D
        - train your machine with the new dataset
    end for
    output is average/vote from all machines trained

- Resulting base-learners are similar because they are drawn from the same original sample
- Resulting base-learners are slightly different due to chance



Bagging

- Not all data points will be used for training
  - Waste of training set
- Bagging is suitable for unstable learning algorithms
  - Unstable algorithms change significantly due to small changes in the data (MLPs, decision trees)
- Example

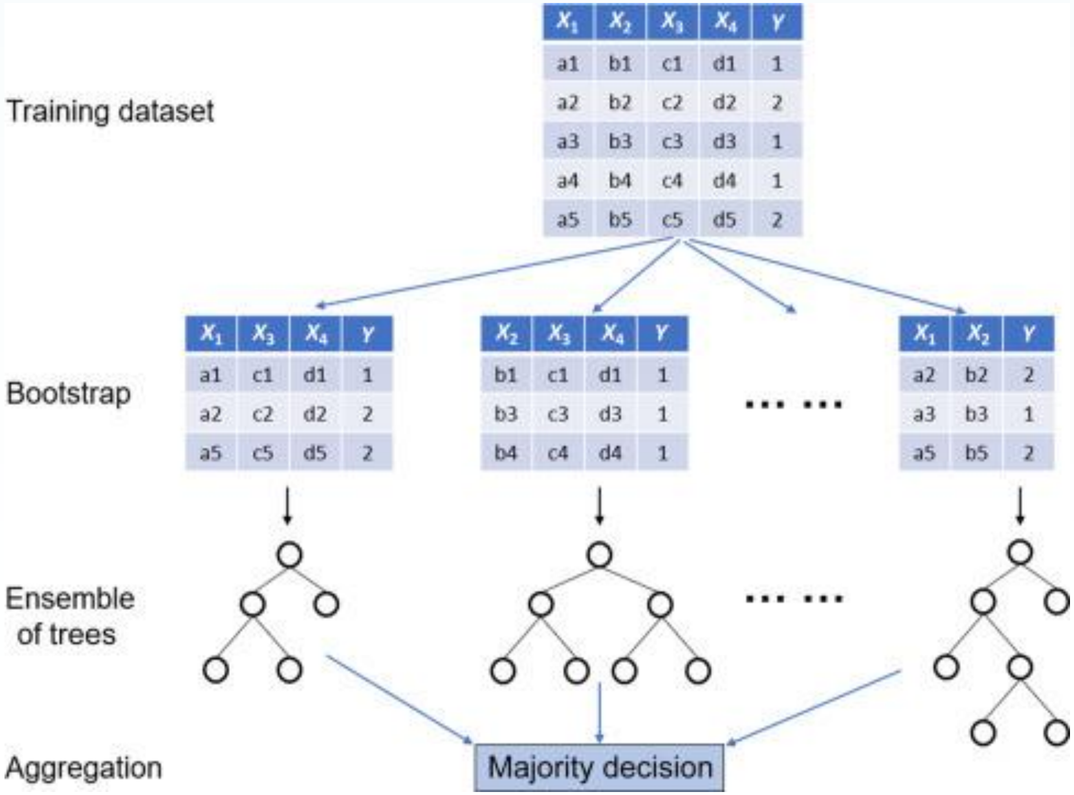
	Single net	*Simple ensemble*	*Bagging*
breast cancer	3.4	3.5	3.4
glass	38.6	35.2	33.1
diabetes	23.9	23	22.8

Error rates *on UCI datasets (10-fold cross validation)*

Source: Opitz & Maclin, 1999

Random forest (decision tree Bagging)

Given a training set S
For i := 1 **to** k **do**:
  Build subset S_i by sampling with replacement from S
  Learn tree T_i from S_i
    At each node:
      Choose best split from **random subset of F features**
    Each tree grows to the largest extent, and no pruning
Make predictions according to majority vote of the set of *k* trees.



Ensemble Methods: Boosting

- In Bagging, generating complementary base-learners is left to chance and unstability of the learning method
- Boosting – Schapire & Freund 1990
  - Try to generate **complementary** weak base-learners by training the next learner on the mistakes of the previous ones
  - **Weak learner**: the learner is required to perform only slightly better than random ($\varepsilon < ½$)
  - **Strong learner**: arbitrary accuracy with high probability
  - Convert a weak learning model to a strong learning model by "boosting" it
    - Kearns and Valient (1988) posed the question, "are the notions of strong and weak learning equivalent?"
    - Schapire (1990) and Freund (1991) gave the first constructive proof

Boosting by Filtering

- Individual experts concentrate on hard-to-learn areas
- Training data for each network comes from a different distribution
- Output of individual networks can be combined by voting or addition (was found to be better in one work)
- Requires large amount of training data
  - Solution: Adaboost (Shapire 1996)
  - Variant of boosting by filtering; short for adaptive boosting

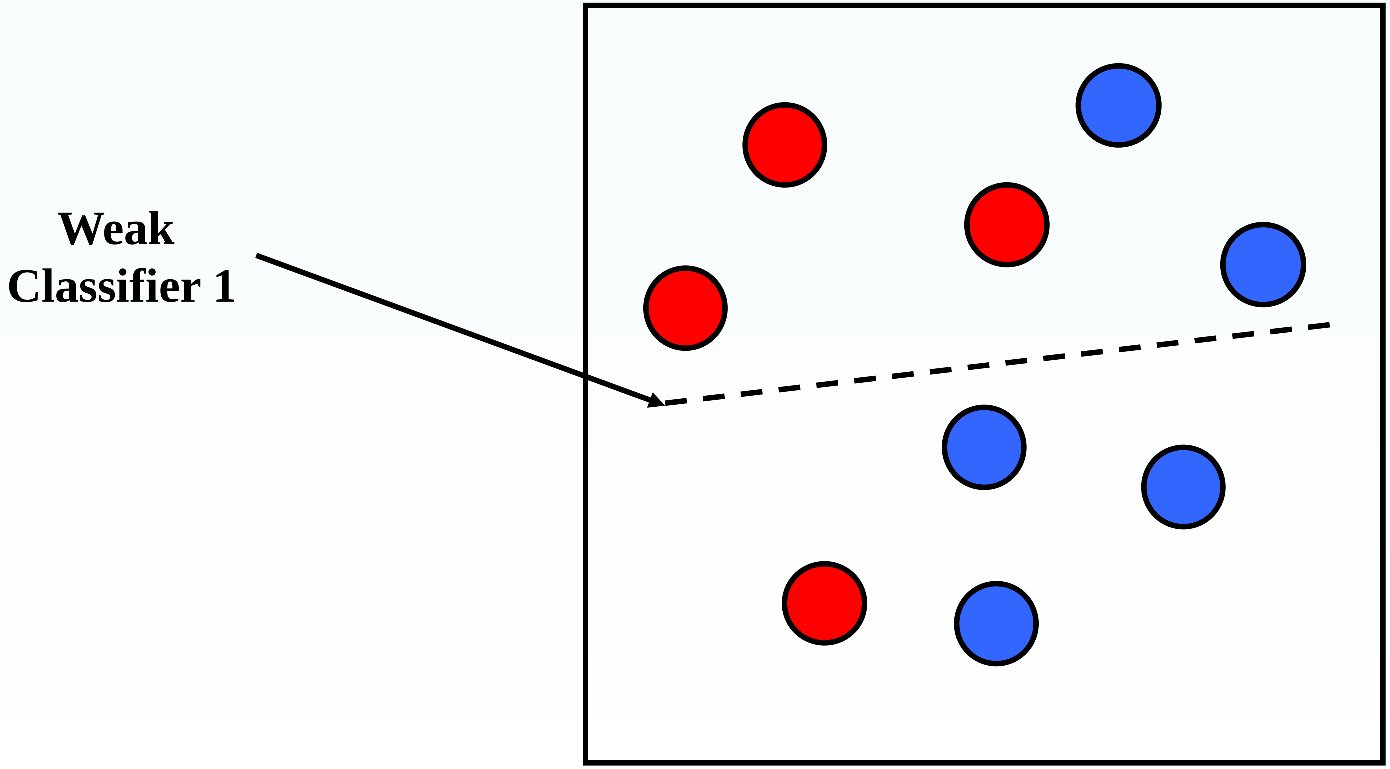
AdaBoost (ADAptive BOOSTing)

- Modify the probabilities of drawing an instance x^t for a classifier j, based on the probability of error of c_j
  - For the next classifier:
    - if pattern x^t is correctly classified, its probability of being selected decreases
    - if pattern x^t is NOT correctly classified, its probability of being selected increases
- All learners must have error less than ½
  - simple, weak learners
  - if not, stop training (the problem gets more difficult for next classifier)

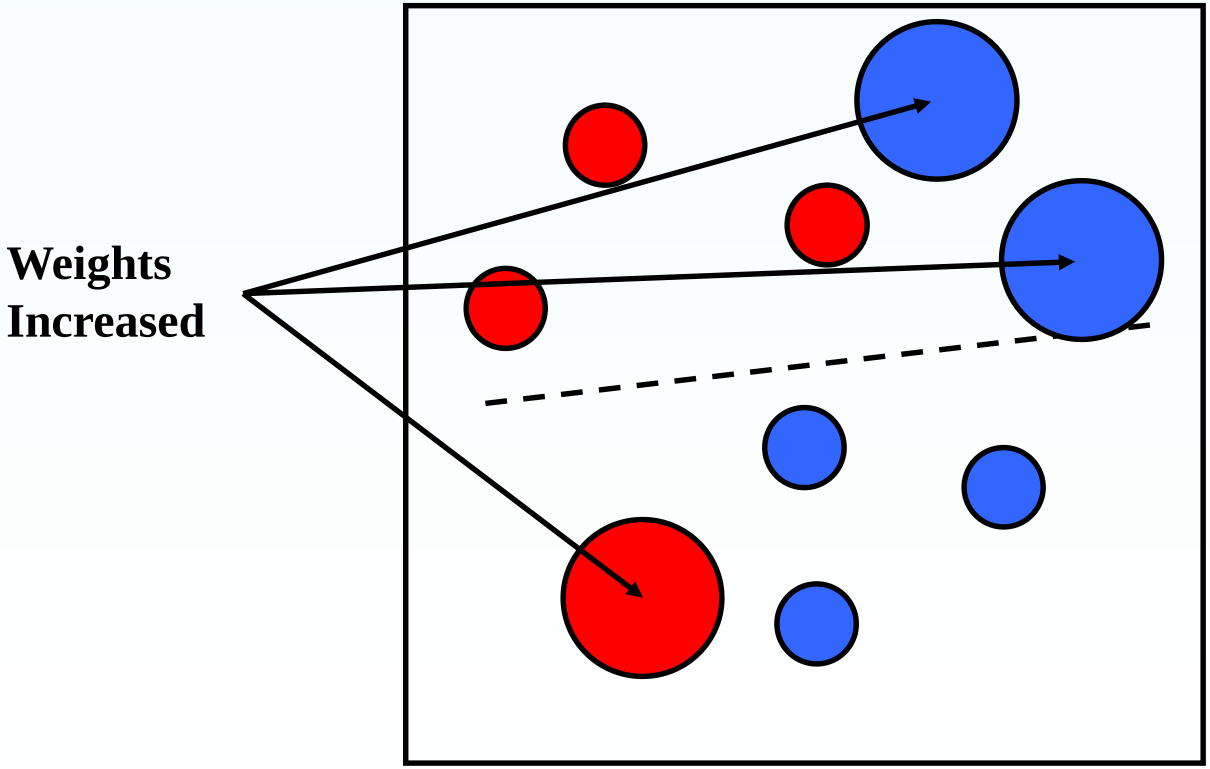
AdaBoost Algorithm

1) The initial distribution is uniform over the training sample.
2) The next distribution is computed by multiplying the weight of example *i* by some number $\beta \in (0,1]$ if the weak hypothesis classifies the input vector correctly; otherwise, the weight is unchanged.
3) The weights are normalized.
4) The final hypothesis is a weighted vote of the L weak classifiers

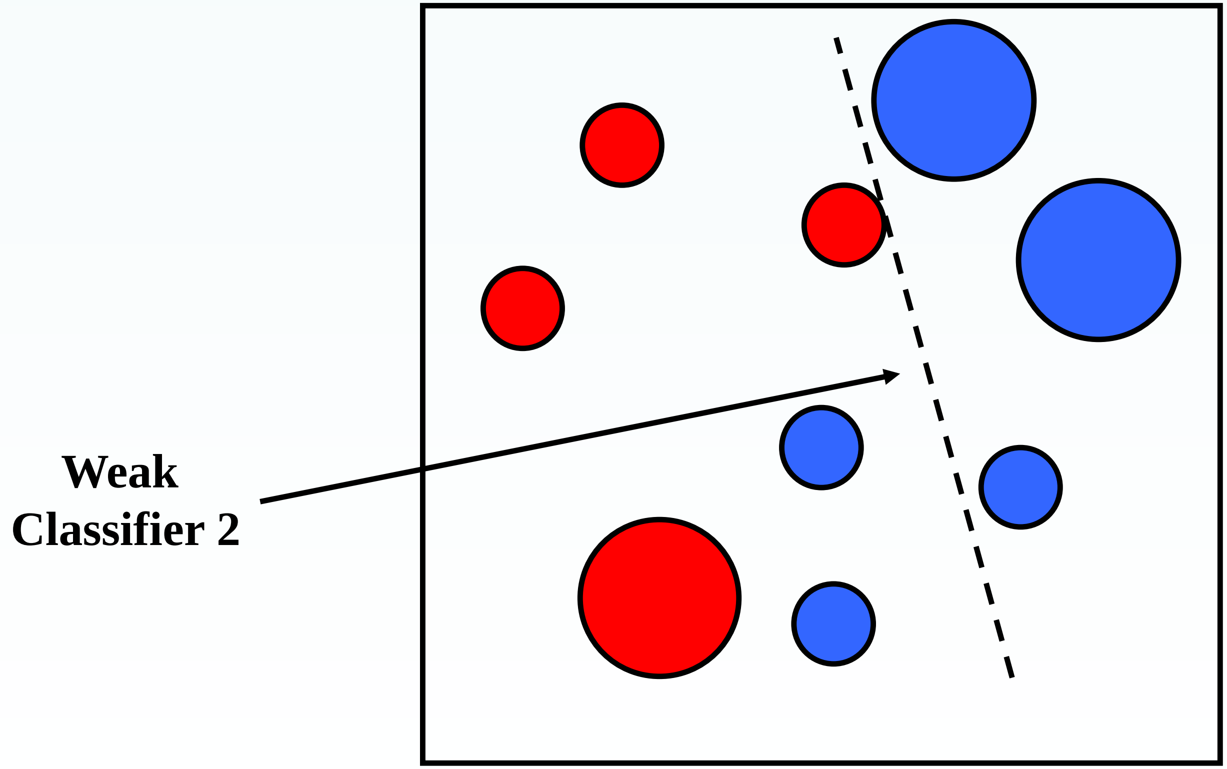
Boosting Illustration



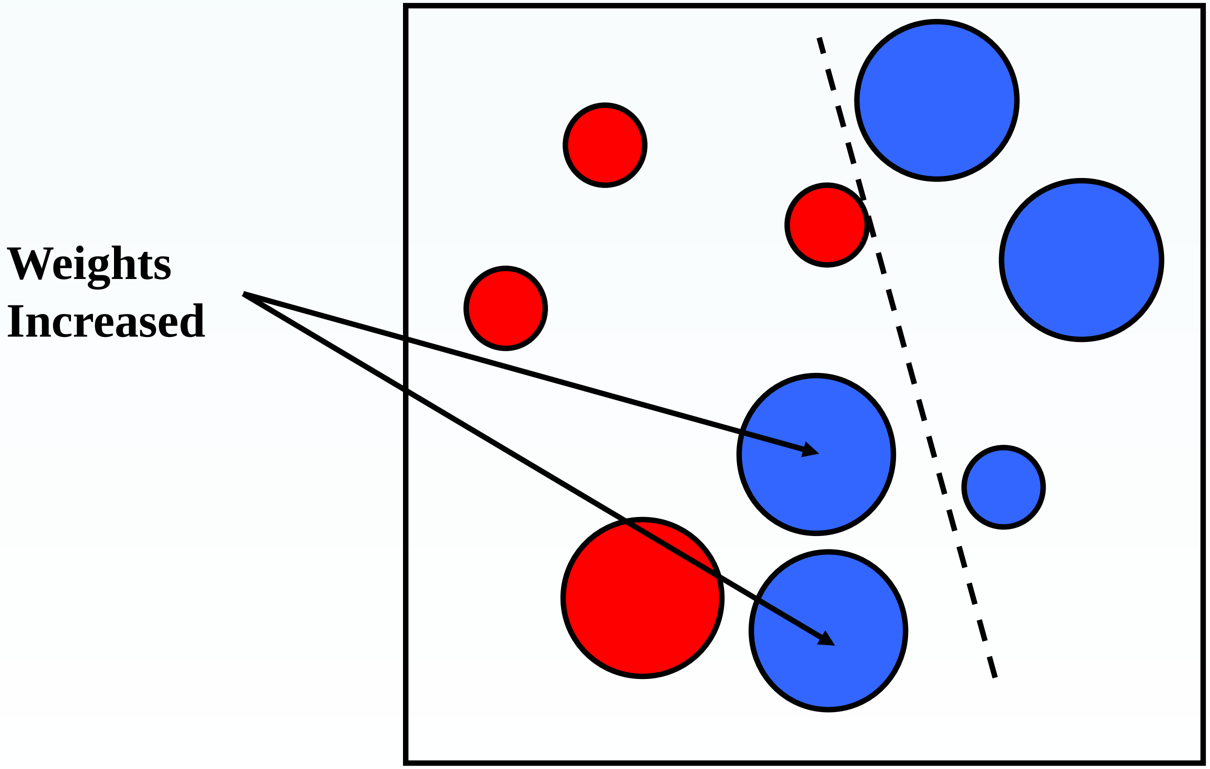
Boosting Illustration



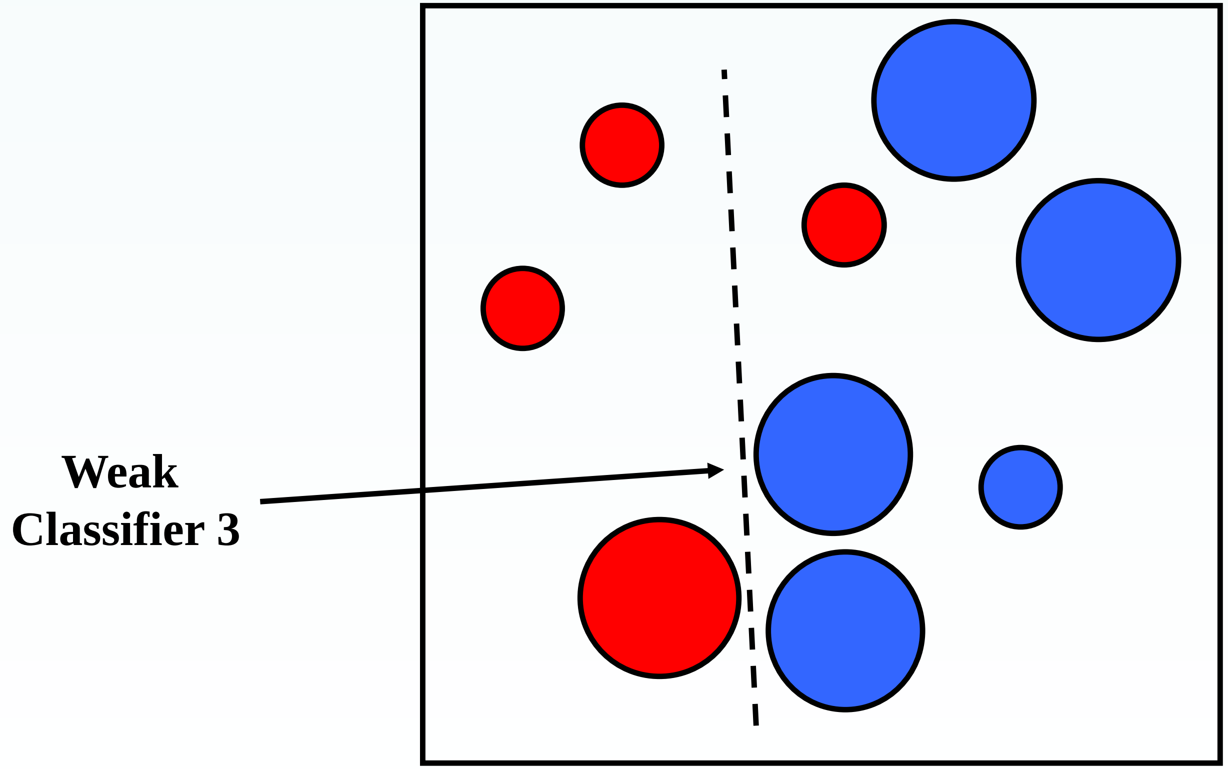
Boosting Illustration



Boosting Illustration

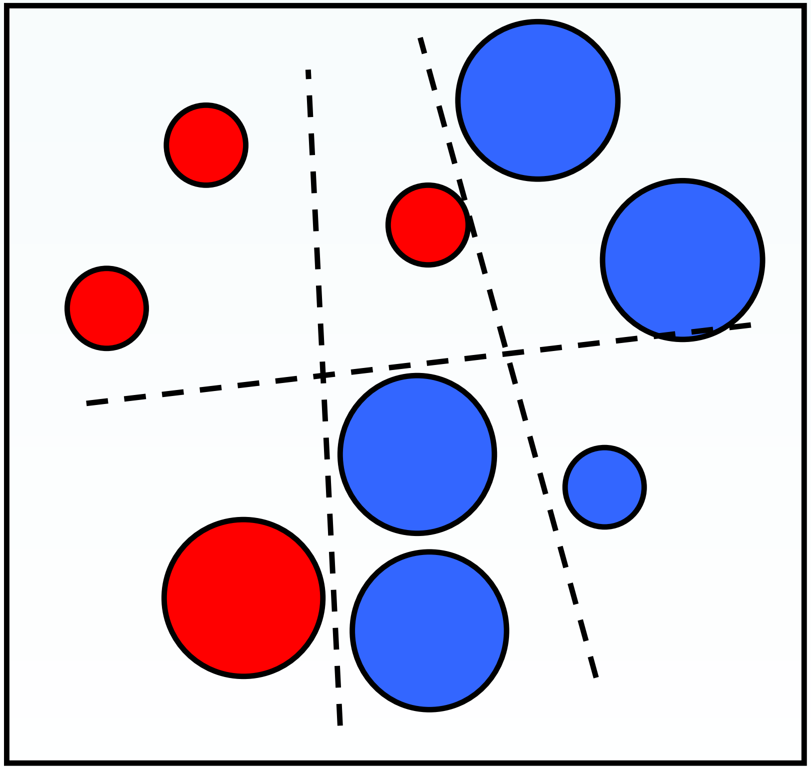


Boosting Illustration



Boosting Illustration

Final classifier is a combination of weak classifiers



Example

	Single net	Simple ensemble	Bagging	AdaBoost
breast cancer	3.4	3.5	3.4	4
glass	38.6	35.2	33.1	31.1
diabetes	23.9	23	22.8	23.3

Error rates on UCI datasets (10-fold cross validation)

Source: Opitz & Maclin, 1999

AdaBoost.M1 algorithm

AdaBoost

For $m = 1$ to M

1. Select and extract from the pool of classifiers the classifier k_m which minimizes

$$W_e = \sum_{y_i \neq k_m(x_i)} w_i^{(m)}$$

2. Set the weight α_m of the classifier to

$$\alpha_m = \frac{1}{2}\ln\left(\frac{1-\mathrm{e_m}}{\mathrm{e_m}}\right)$$

   where $e_m = W_e/W$

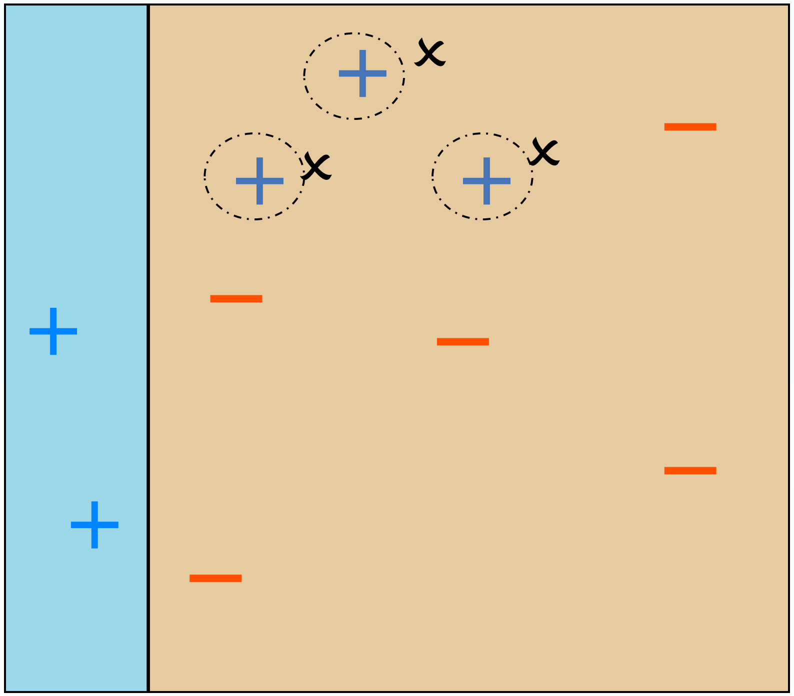
3. Update the weights of the data points for the next iteration. If $k_m(x_i)$ is a miss, set

$$w_i^{(m+1)} = w_i^{(m)} \mathrm{e}^{\alpha_m} = w_i^{(m)} \sqrt{\frac{1-e_m}{e_m}}$$

   otherwise

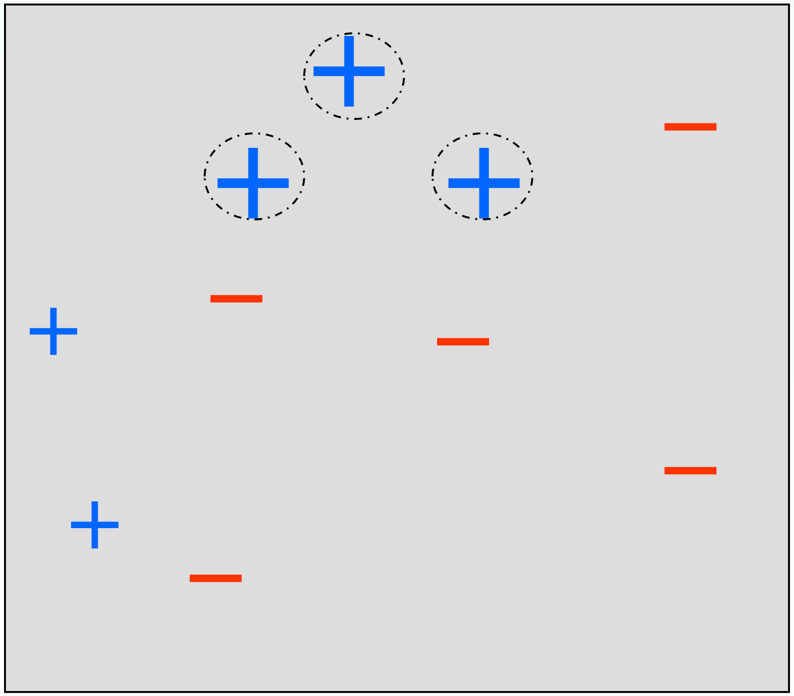
$$w_i^{(m+1)} = w_i^{(m)} \mathrm{e}^{-\alpha_m} = w_i^{(m)} \sqrt{\frac{e_m}{1-e_m}}$$

Round 1 of 3



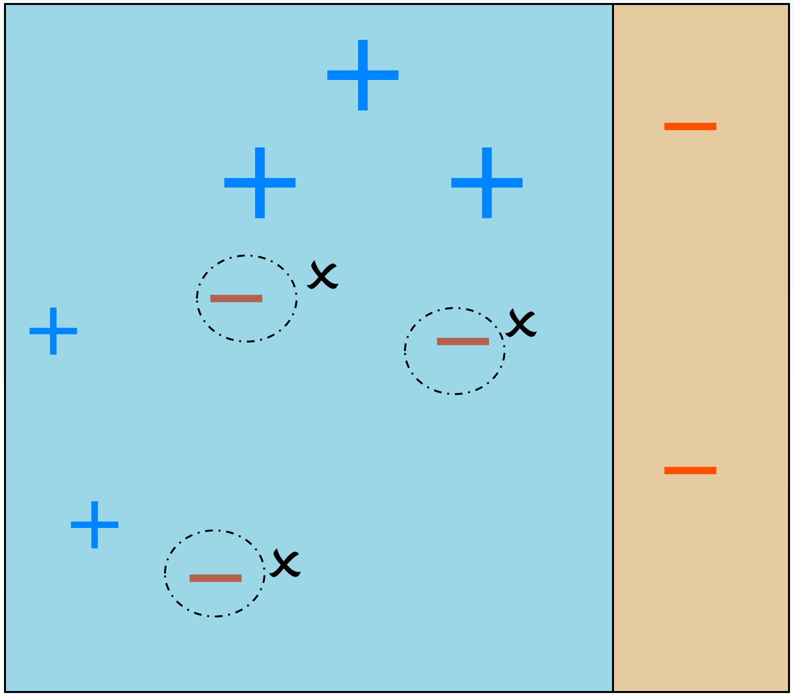
h_1 $\varepsilon_1 = 0.300$

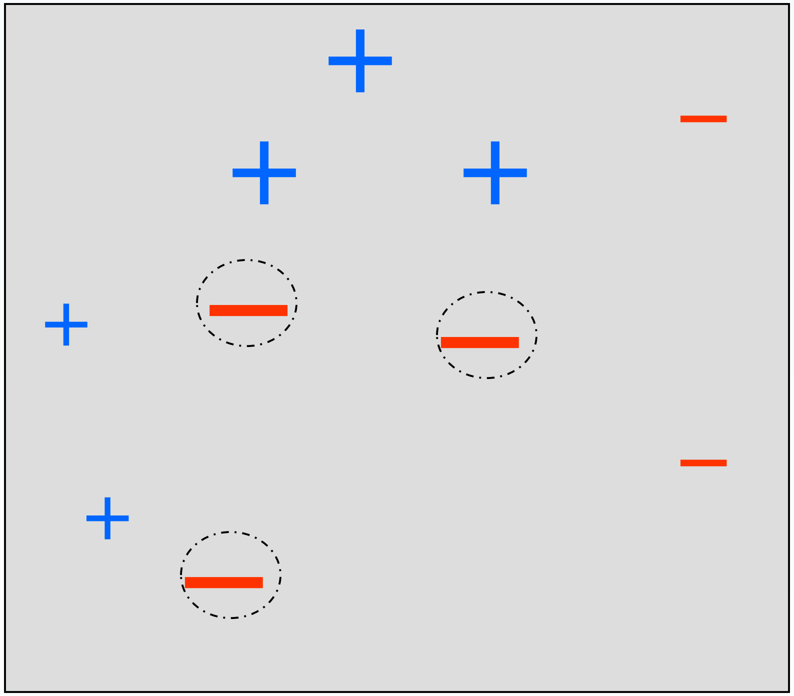
α_1=0.424



D_2

Round 2 of 3

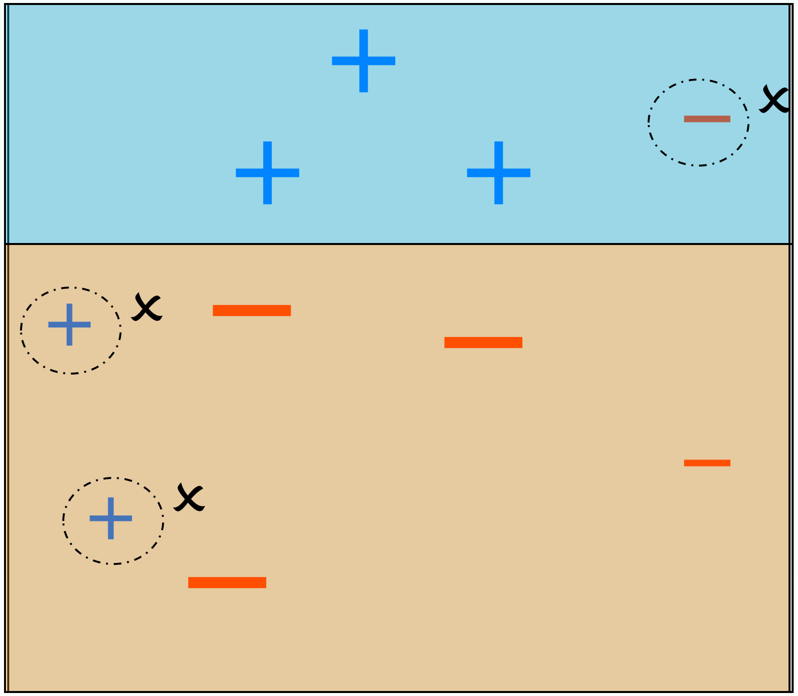




$\varepsilon_2 = 0.196$ h_2 D_2

$\alpha_2 = 0.704$

Round 3 of 3



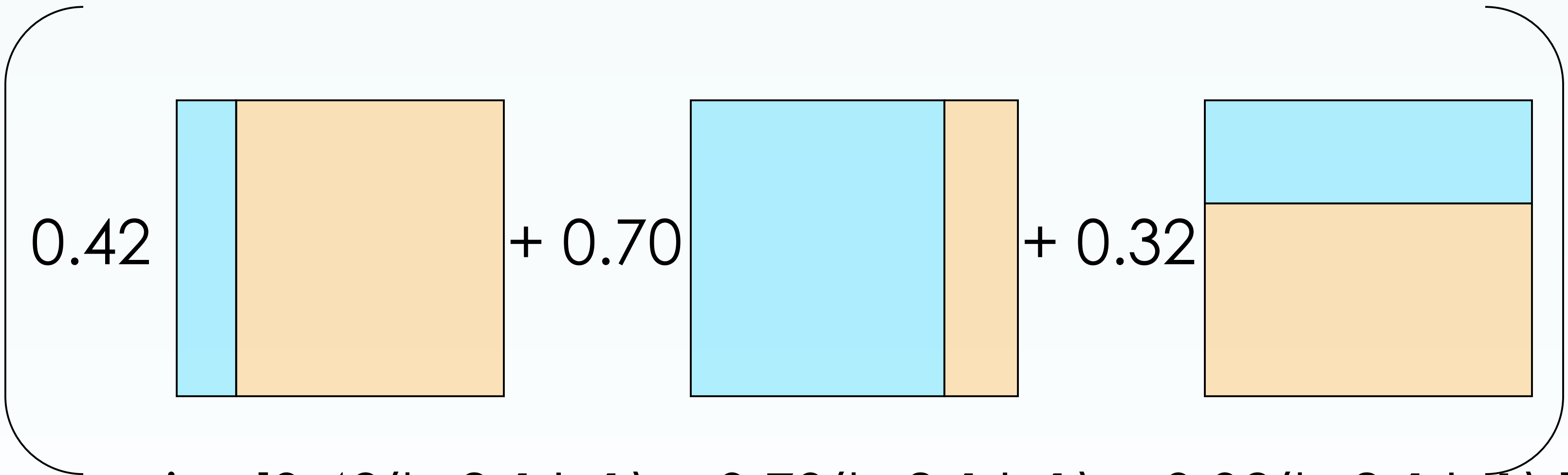
h_3

STOP

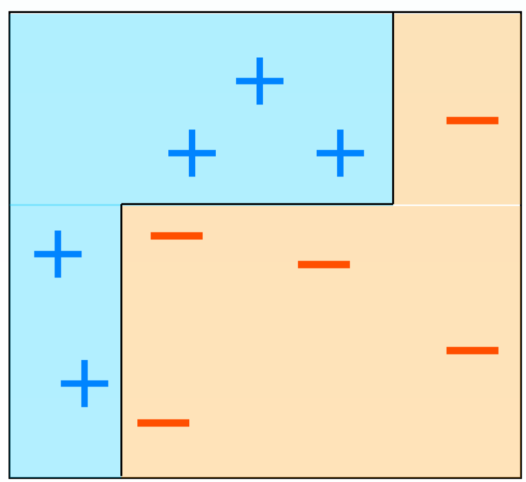
$\varepsilon_3 = 0.344$

α_2=0.323

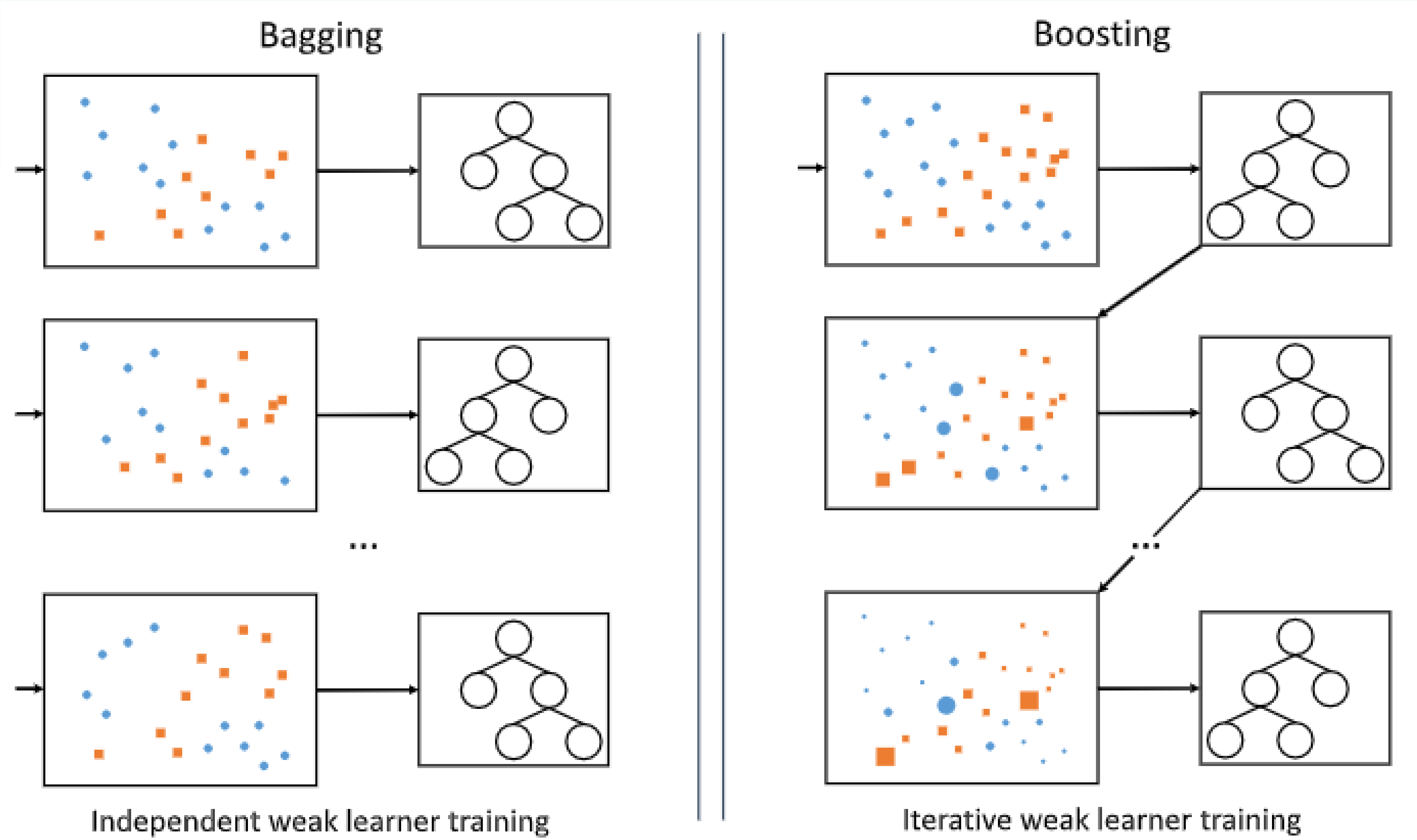
Final Hypothesis



$$H_{final} = \text{sign}[0.42(h_1 ?\ 1\,|\,{-1}) + 0.70(h_2 ?\ 1\,|\,{-1}) + 0.32(h_3 ?\ 1\,|\,{-1})\ ]$$



Bagging vs. Boosting



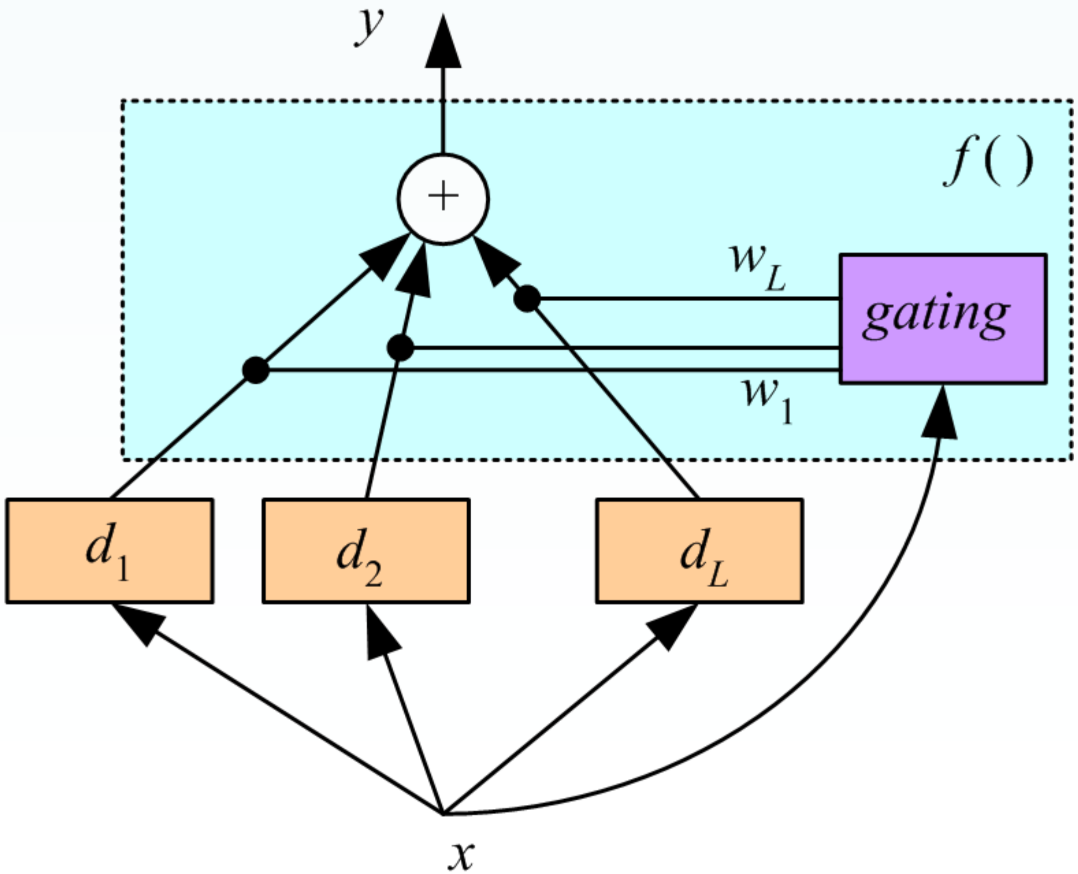
Dynamic Methods: Mixture of Experts

Voting where weights are input-dependent (gating) not constant

$$g_i(x) = \frac{\exp(\gamma_i)}{\sum_{j=1} \exp(\gamma_j)}$$

$$y = \sum_{j=1}^{L} w_j d_j$$

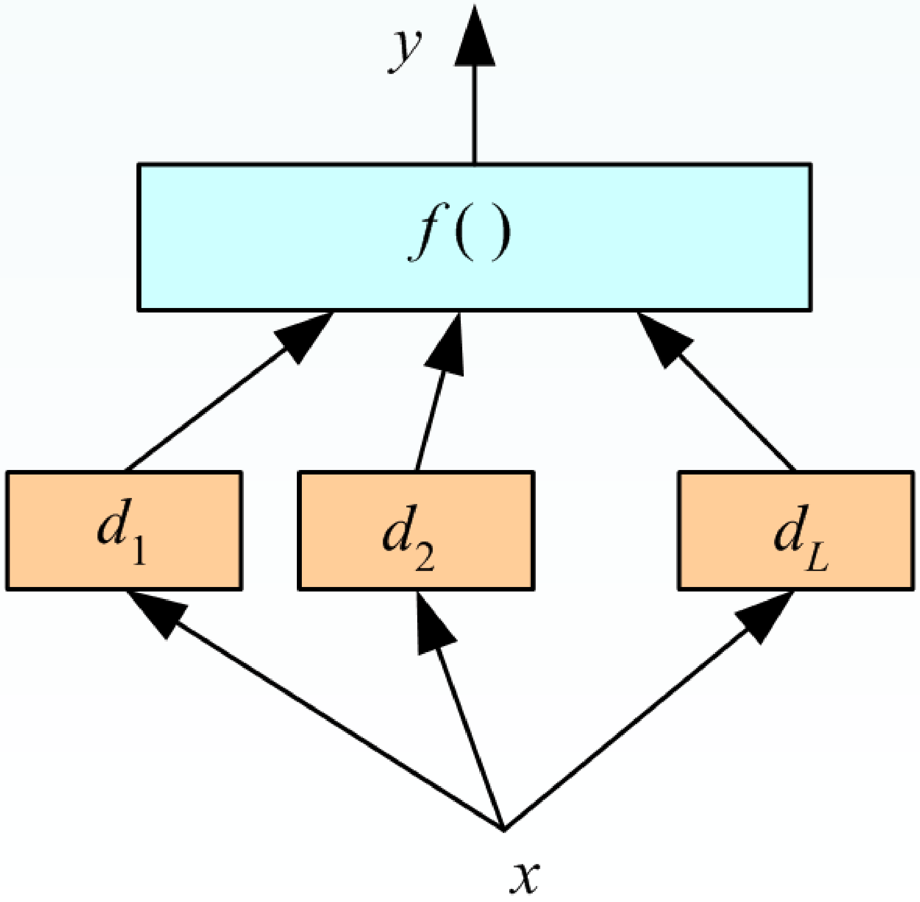
(Jacobs et al., 1991)



- In general, experts or gating can be non-linear
- Base learners become experts in diff. parts of the input space

Dynamic Methods: Stacking

- Combiner **f()** is another learner (Wolpert, 1992)



- f() need not be linear, it can be a neural network
- We cannot train f() on the training data; combiner should learn how the baselearners make errors.
  - Leave-one-out or kfold cross validation
- Learners should be as different as possible, to complement each other ideally using different learning algorithms

General Rules of Thumb

- Components should exhibit low correlation - understood well for regression, not so well for classification. "Overproduce-and-choose" is a good strategy.
- Unstable estimators (e.g. NNs, decision trees) benefit most from ensemble methods. Stable estimators like k-NN tend not to benefit.
- Boosting tends to suffer on noisy data.
- Techniques manipulate either training data, architecture of learner, initial configuration, or learning algorithm. Training data is seen as most successful route; initial configuration is least successful.
- Uniform weighting is almost never optimal. Good strategy is to set the weighting for a component proportional to its error on a validation set.

Reading

- E. Alpaydin, **Introduction to Machine Learning**, 3rd ed., The MIT Press, 2014. (ch. 17)
- S. Haykin, **Neural Networks: A Comprehensive Foundation,** Prentice Hall, 1998. (ch. 7)
- C. M. Bishop, **Pattern recognition and machine learning**, Springer, 2006. (ch. 14)
- R. E. Schapire, **The boosting approach to machine learning: An overview**. Nonlinear estimation and classification, pp. 149-171, 2003.