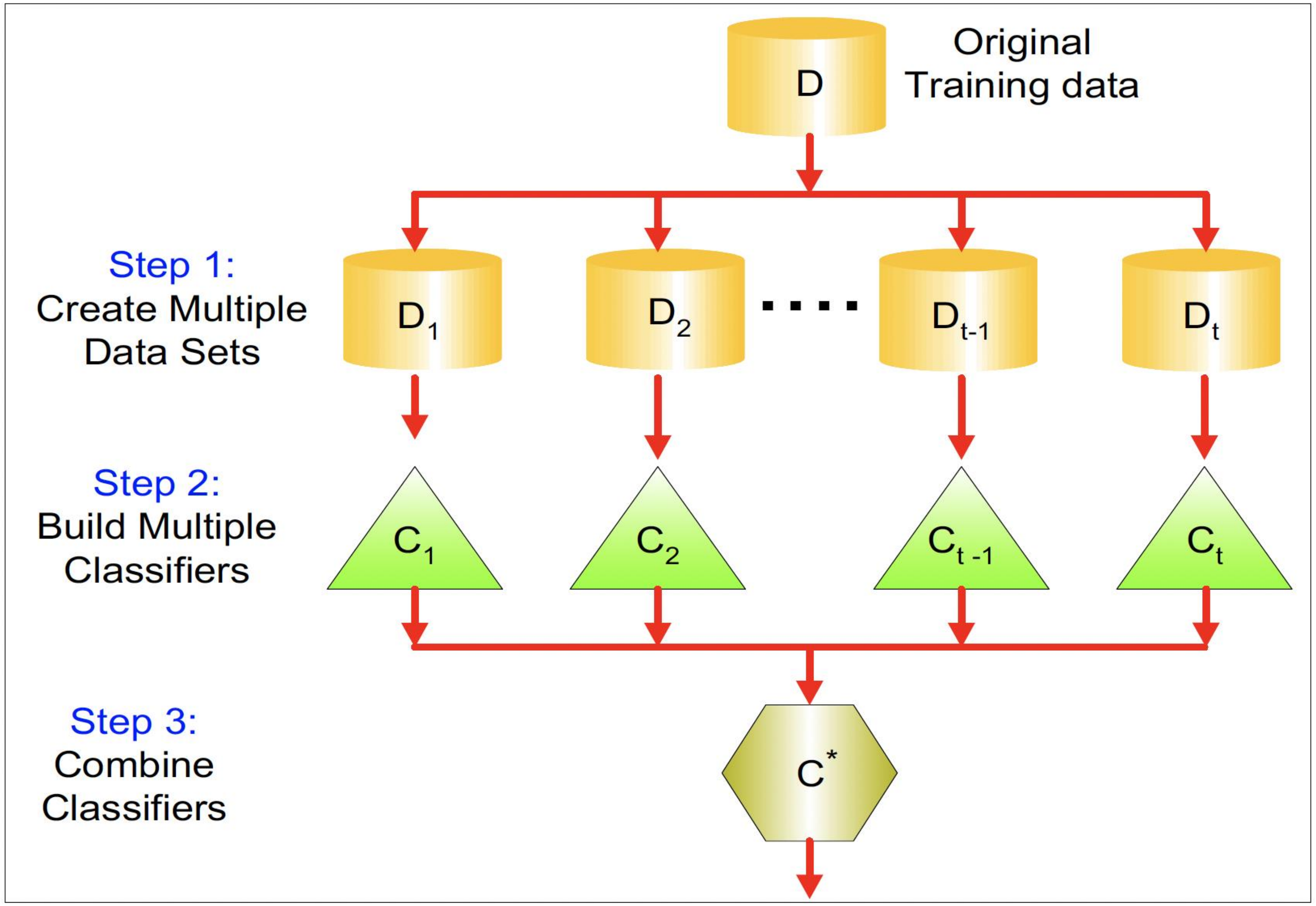
Ensemble Methods

DSBA/MBAD 6211

Ensemble Methods

- Also called **classifier combination** methods
- Techniques for improving classification accuracy by aggregating the predictions of multiple classifiers
- Constructs a set of **base classifiers** from training data and performs classification by taking a vote on the predictions made by each base classifier

General Idea


D
Original Training data
Step 1: Create Multiple Data Sets
D_1
D_2
....
D_{t-1}
D_t
Step 2: Build Multiple Classifiers
C_1
C_2
C_{t-1}
C_t
Step 3: Combine Classifiers
C^*

How does it work?

Assume we have **25** base classifiers:

- Each classifier has error rate, $\boldsymbol{\varepsilon = 0.35}$
- Assume classifiers are independent
- Probability that the ensemble classifier makes a wrong prediction is:

?

How does it work?

Assume we have **25** base classifiers:

- Each classifier has error rate, $\boldsymbol{\varepsilon = 0.35}$
- Assume classifiers are independent
- Probability that the ensemble classifier makes a wrong prediction is:

$$e_{ensemble} = \sum_{i=13}^{25} \binom{25}{i} e^{i}(1-\varepsilon)^{25-i} = 0.06$$

How does it work?

Assume we have **25** base classifiers:

- Each classifier has error rate, $\boldsymbol{\varepsilon = 0.35}$
- Assume classifiers are independent
- Probability that the ensemble classifier makes a wrong prediction is:

$$e_{ensemble} = \sum_{i=13}^{25} \binom{25}{i} e^{i}(1-\varepsilon)^{25-i} = 0.06$$

- There are two necessary conditions for an ensemble classifier to perform better than a single classifier:
    1. The base classifier should be independent of each other.
    2. The base classifier should do better than a classifier that performs random guessing.

Ensemble Classifier Construction

1. By manipulating the training set
2. By manipulating the training features
3. By manipulating the class labels
4. By manipulating the leaning algorithm

1,2 and 3 are generic methods applicable to any classifier, 4 depends on the type of the classifier used.

Manipulating the training set

- Multiple training sets are created by resampling the original data according to some sampling distribution.
- A classifier is then built from each training set using a particular learning algorithm examples include (**bagging** and **boosting**).

Manipulating the training features

- A subset of input features is chosen to form each training set.
- The subset can be either chosen randomly or based on recommendation of domain experts.
- Example include **Random Forest**.

Manipulating the class labels

- Used when the number of classes is sufficiently large.
- The training data is transformed into a binary class problem by randomly partitioning the class labels into two disjoint subsets, A_0 and A_1.
- Some classes are assigned to subset A_0 and belong to class 0 and the remaining classes are assigned to set A_1 and belong to class 1.
- The labeled examples are then used to train a base classifier. By repeating the class-relabeling and model-building steps multiple times, an ensemble of base classifiers is obtained.
- Example includes error correcting code

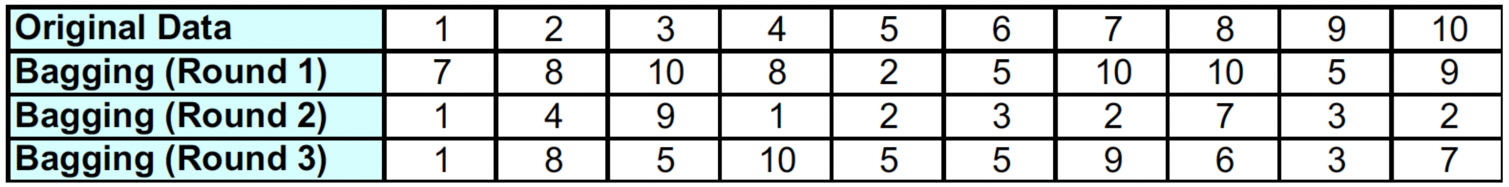
Manipulating the leaning algorithm

- Many learning algorithms can be manipulated in such a way that applying the algorithm several times on the same training data may result in different models.
- For example, an artificial neural network can produce different models by changing its network topology or the initial weights of the links between neurons.
- Similarly, an ensemble of decision trees can be constructed by injecting randomness into the tree-growing procedure. For example, instead of choosing the best splitting attribute at each node, we can randomly choose one of the top k attributes for splitting

Bagging

Bagging, also known as **bootstrap aggregating**, is a technique that repeatedly samples (with replacement) from a data set according to a uniform probability distribution. Each bootstrap sample has the same size as the original data. Because the sampling is done with replacement, some instances may appear several times in the same training set, while others may be omitted from the training set. On average, a bootstrap sample D_i contains approximately *63%* of the original training data.

Each sample has a probability $1-(1-1/N)^N$ in each D_i

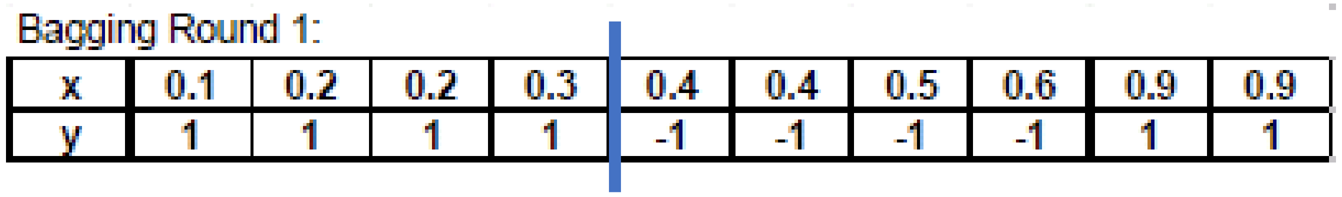


Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

Bagging

Data (*x*)	x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Label (*y*)	y	1	1	1	-1	-1	-1	-1	1	1	1

sample



Bagging Round 1:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

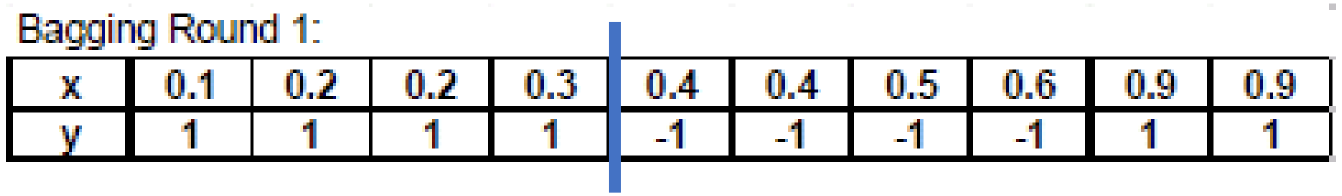
x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Weak classifier: 1-level binary decision tree - decision stump

Bagging

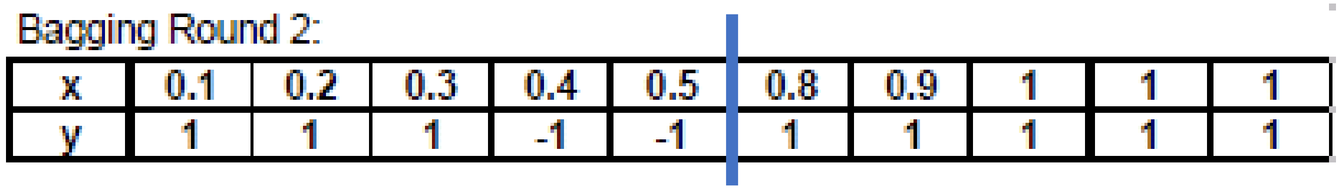
Data (*x*)	x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Label (*y*)	y	1	1	1	-1	-1	-1	-1	1	1	1

sample


Bagging Round 1:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1


Bagging Round 2:

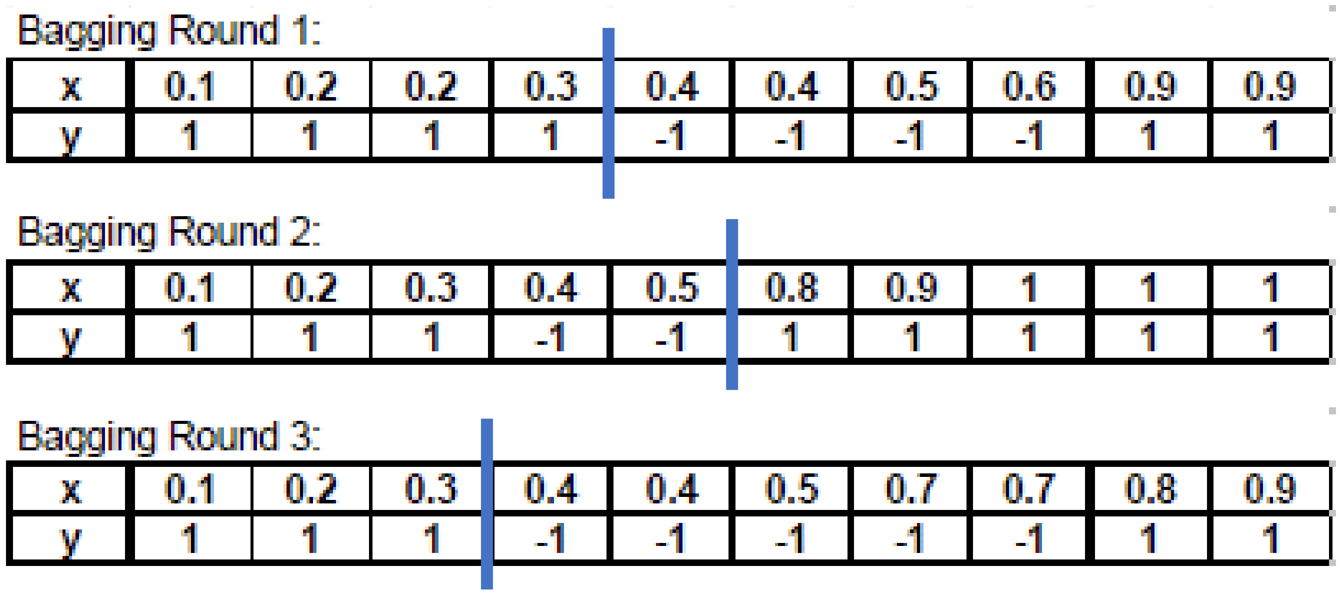
x	0.1	0.2	0.3	0.4	0.5	0.8	0.9	1	1	1
y	1	1	1	-1	-1	1	1	1	1	1

x <= 0.65 ==> y = 1
x > 0.65 ==> y = 1

Bagging

Data (*x*) x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Label (*y*) y	1	1	1	-1	-1	-1	-1	1	1	1

sample



Bagging Round 1:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging Round 2:

x	0.1	0.2	0.3	0.4	0.5	0.8	0.9	1	1	1
y	1	1	1	-1	-1	1	1	1	1	1

x <= 0.65 ==> y = 1
x > 0.65 ==> y = 1

Bagging Round 3:

x	0.1	0.2	0.3	0.4	0.4	0.5	0.7	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging

Bagging Round 4:

x	0.1	0.1	0.2	0.4	0.4	0.5	0.5	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.3 ==> y = 1
x > 0.3 ==> y = -1

Bagging Round 5:

x	0.1	0.1	0.2	0.5	0.6	0.6	0.6	1	1	1
y	1	1	1	-1	-1	-1	-1	1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging Round 6:

x	0.2	0.4	0.5	0.6	0.7	0.7	0.7	0.8	0.9	1
y	1	-1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 7:

x	0.1	0.4	0.4	0.6	0.7	0.8	0.9	0.9	0.9	1
y	1	-1	-1	-1	-1	1	1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 8:

x	0.1	0.2	0.5	0.5	0.5	0.7	0.7	0.8	0.9	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 9:

x	0.1	0.3	0.4	0.4	0.6	0.7	0.7	0.8	1	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

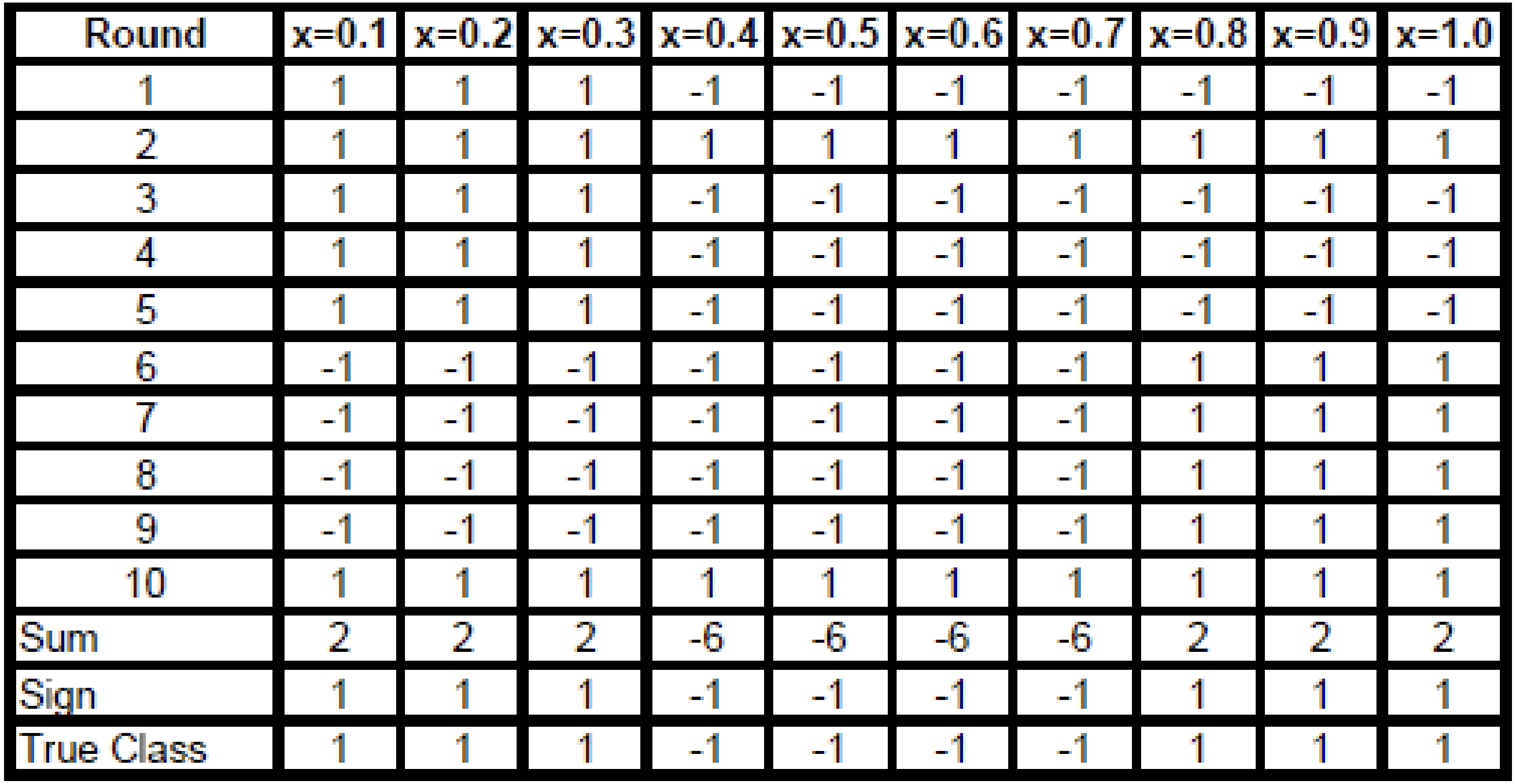
x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 10:

x	0.1	0.1	0.1	0.1	0.3	0.3	0.8	0.8	0.9	0.9
y	1	1	1	1	1	1	1	1	1	1

x <= 0.05 ==> y = -1
x > 0.05 ==> y = 1

Bagging



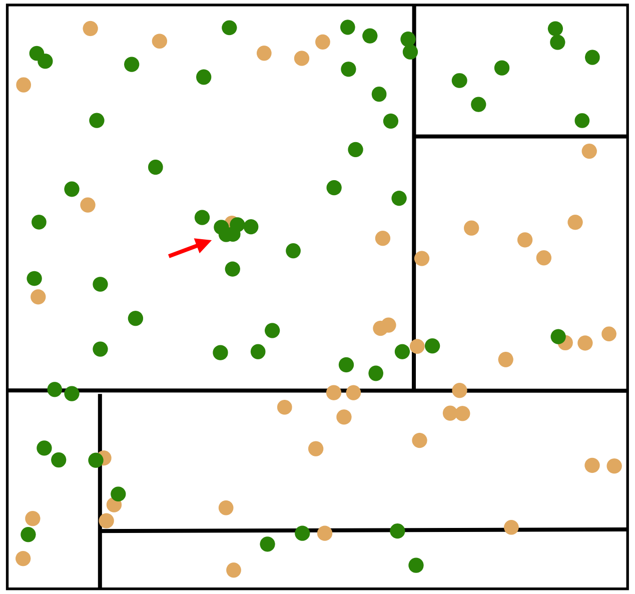
Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	1	1	1	-1	-1	-1	-1	-1	-1	-1
2	1	1	1	1	1	1	1	1	1	1
3	1	1	1	-1	-1	-1	-1	-1	-1	-1
4	1	1	1	-1	-1	-1	-1	-1	-1	-1
5	1	1	1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	1	1	1
7	-1	-1	-1	-1	-1	-1	-1	1	1	1
8	-1	-1	-1	-1	-1	-1	-1	1	1	1
9	-1	-1	-1	-1	-1	-1	-1	1	1	1
10	1	1	1	1	1	1	1	1	1	1
Sum	2	2	2	-6	-6	-6	-6	2	2	2
Sign	1	1	1	-1	-1	-1	-1	1	1	1
True Class	1	1	1	-1	-1	-1	-1	1	1	1

A Random Forest

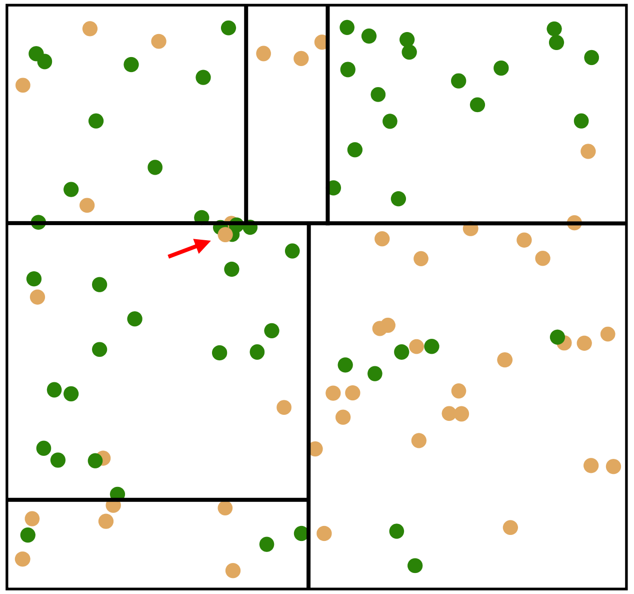
- ❖An ensemble model is an aggregation of more than one model where the final prediction of the model is a combination of the predictions from the component models of the ensemble.
- ❖A random forest model is an ensemble of classification or regression trees.
- ❖The forest models were developed to overcome the instability that a single classification or regression tree exhibits with minor perturbations of the training data.

Visualization: https://towardsdatascience.com/why-random-forests-outperform-decision-trees-1b0f175a0b5

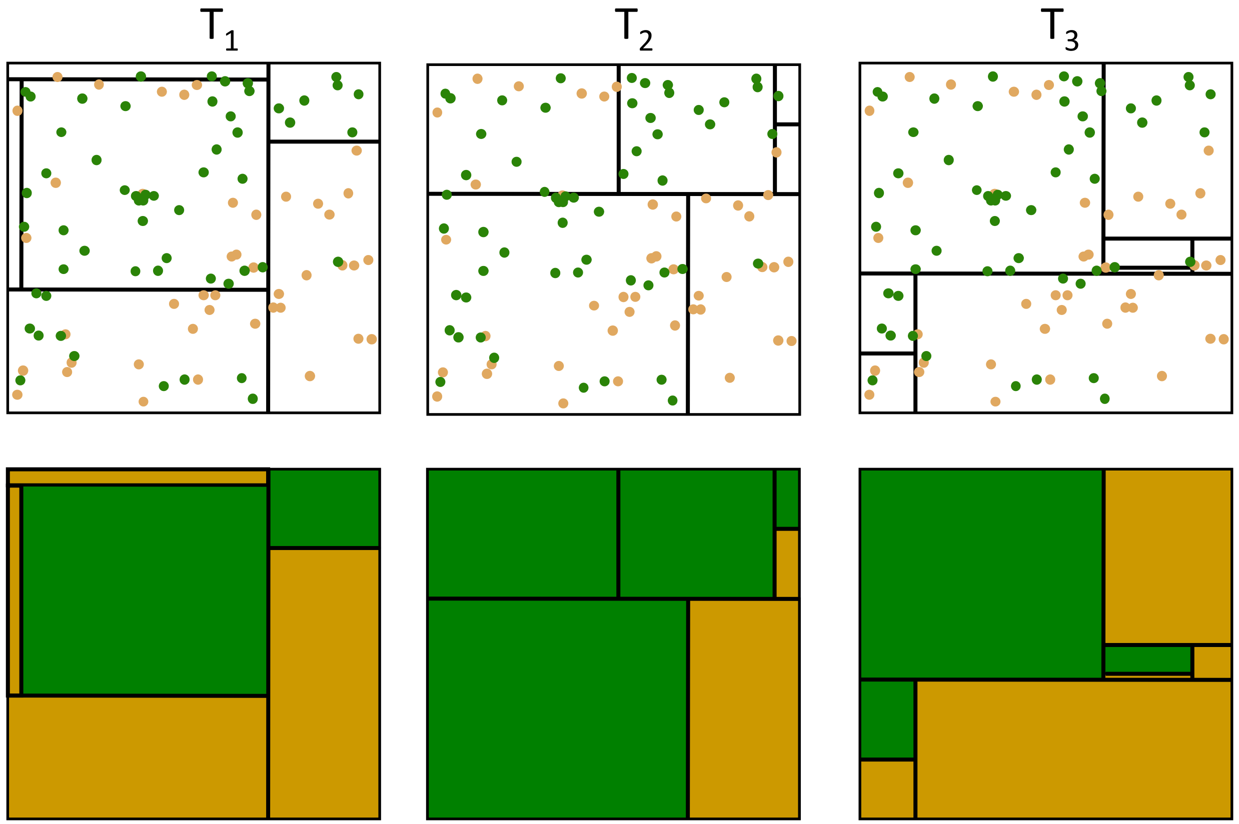
Instability


Accuracy = 81%

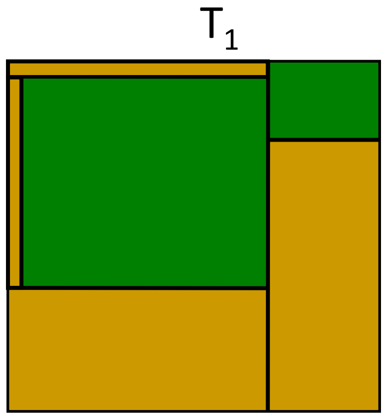
One reversal

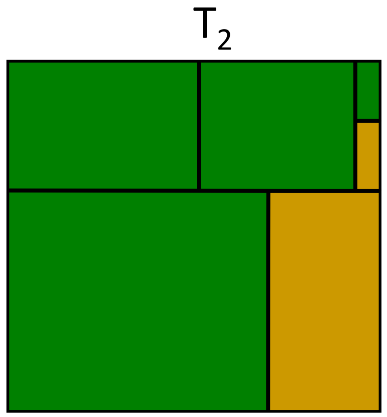

Accuracy = 80%

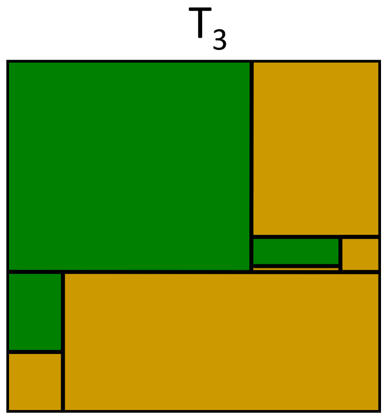
Perturb



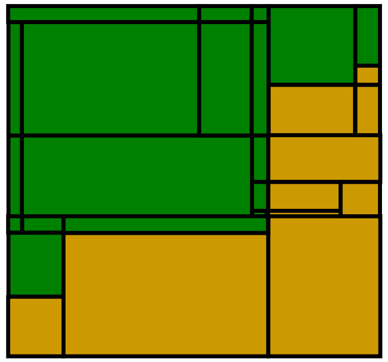
Combine







ave(T_1, T_2, T_3) =



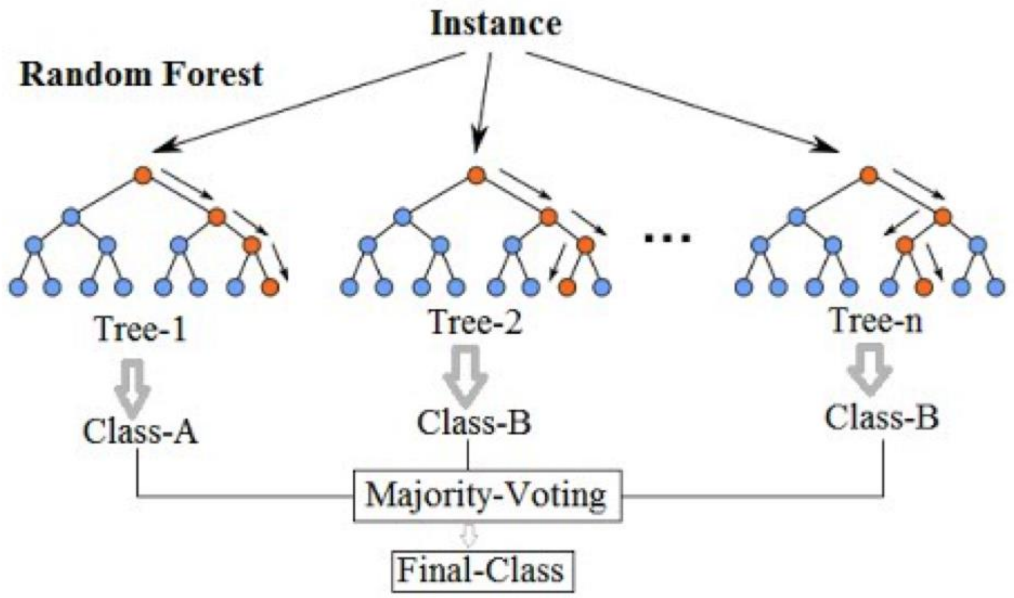
Random Forest

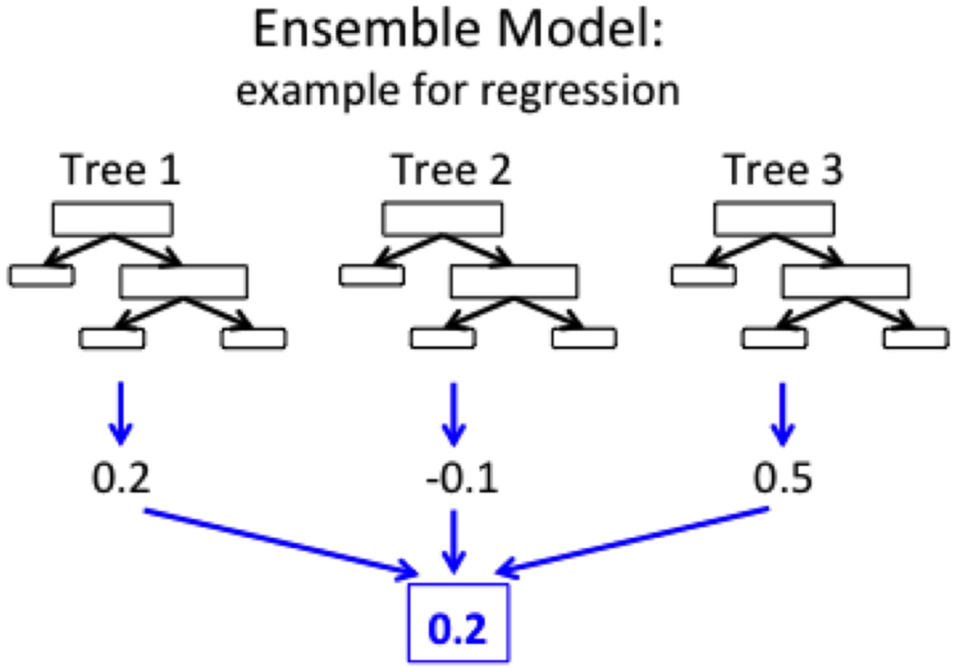
- *Why is it called random in 'Random Forest'?*

- "Random" refers to mainly two processes:
  1. random observations to grow each tree and
  2. random variables selected for splitting at each node (e.g., sqrt of number of variables).

How Random Forest Works

- ❖Random record selection
- ❖Random variable selection (say, m)
  - Classification model: m is the square root of the total number of predictors
  - Regression model: m is one third of the total number of predictors
- ❖For each tree, calculate the error rate
- ❖Aggregate error from all trees to determine the RF performance
- ❖Determine the results by averaging votes over all trees in the forest


Instance
Random Forest
Tree-1
Tree-2
...
Tree-n
Class-A
Class-B
Class-B
Majority-Voting
Final-Class


Ensemble Model:
example for regression
Tree 1
Tree 2
Tree 3
0.2
-0.1
0.5
0.2

Boosting

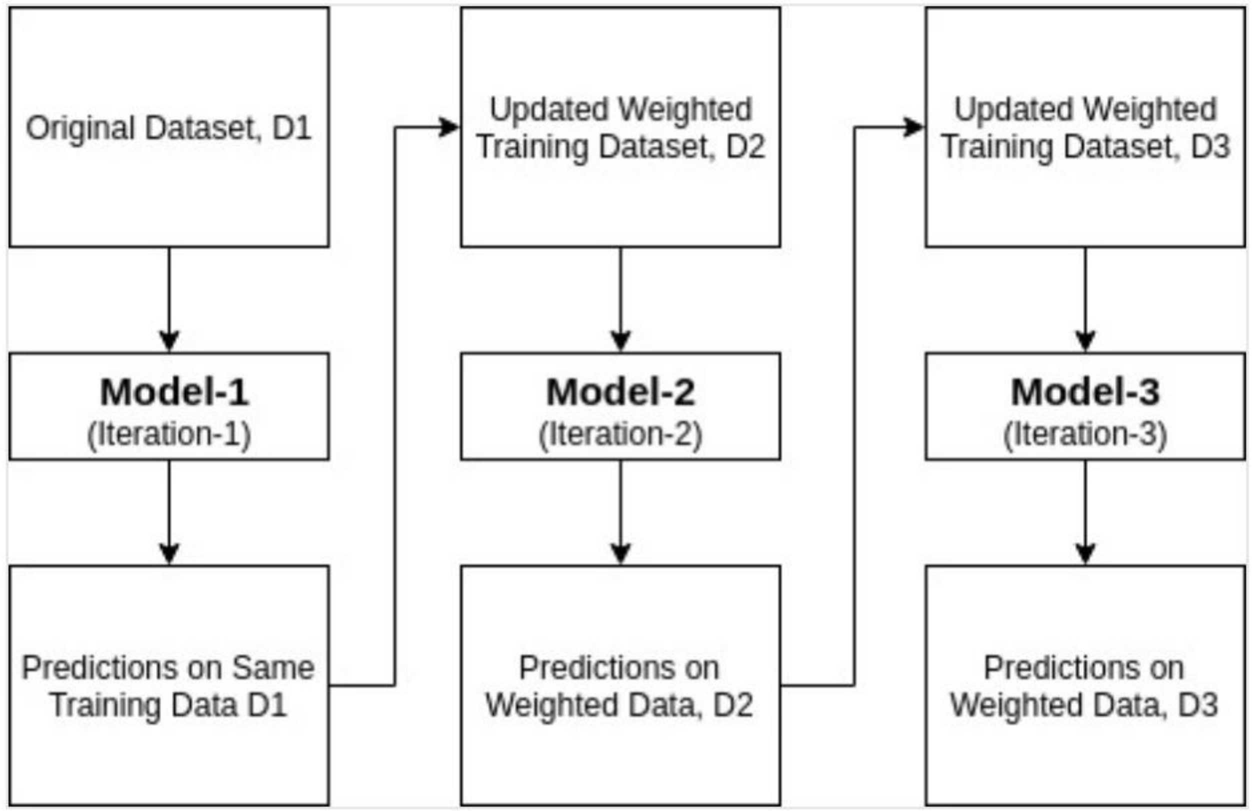
Boosting is an iterative procedure used to adaptively change the distribution of training examples so that the base classifiers will focus on examples that are hard to classify. Unlike bagging, boosting assigns a weight to each training example and may adaptively change the weight at the end of each boosting round. The weights assigned to the training examples can be used in the following ways:

1. They can be used as a sampling distribution to draw a set of bootstrap samples from the original data.
2. They can be used by the base classifier to learn a model that is biased toward higher-weight examples.

Boosting

- Boosting converts the “weak” learning algorithm into the strong one
- Initially all N records are assigned the same weight 1/N
- Unlike in bagging, in boosting the weights may change at the end of the run
- An iterative technique to adaptively change distribution of training data by focusing more on previously misclassified records
- Records that are wrongly classified will have their weights increased
- Records that are classified correctly will have their weights decreased

Boosting


Original Dataset, D1
Model-1
(Iteration-1)
Predictions on Same Training Data D1
Updated Weighted Training Dataset, D2
Model-2
(Iteration-2)
Predictions on Weighted Data, D2
Updated Weighted Training Dataset, D3
Model-3
(Iteration-3)
Predictions on Weighted Data, D3

AdaBoost

Evaluate:

$$\varepsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I(f_m(x_n) \neq y_n)}{\sum_{n=1}^{N} w_n^{(m)}}$$

And then:

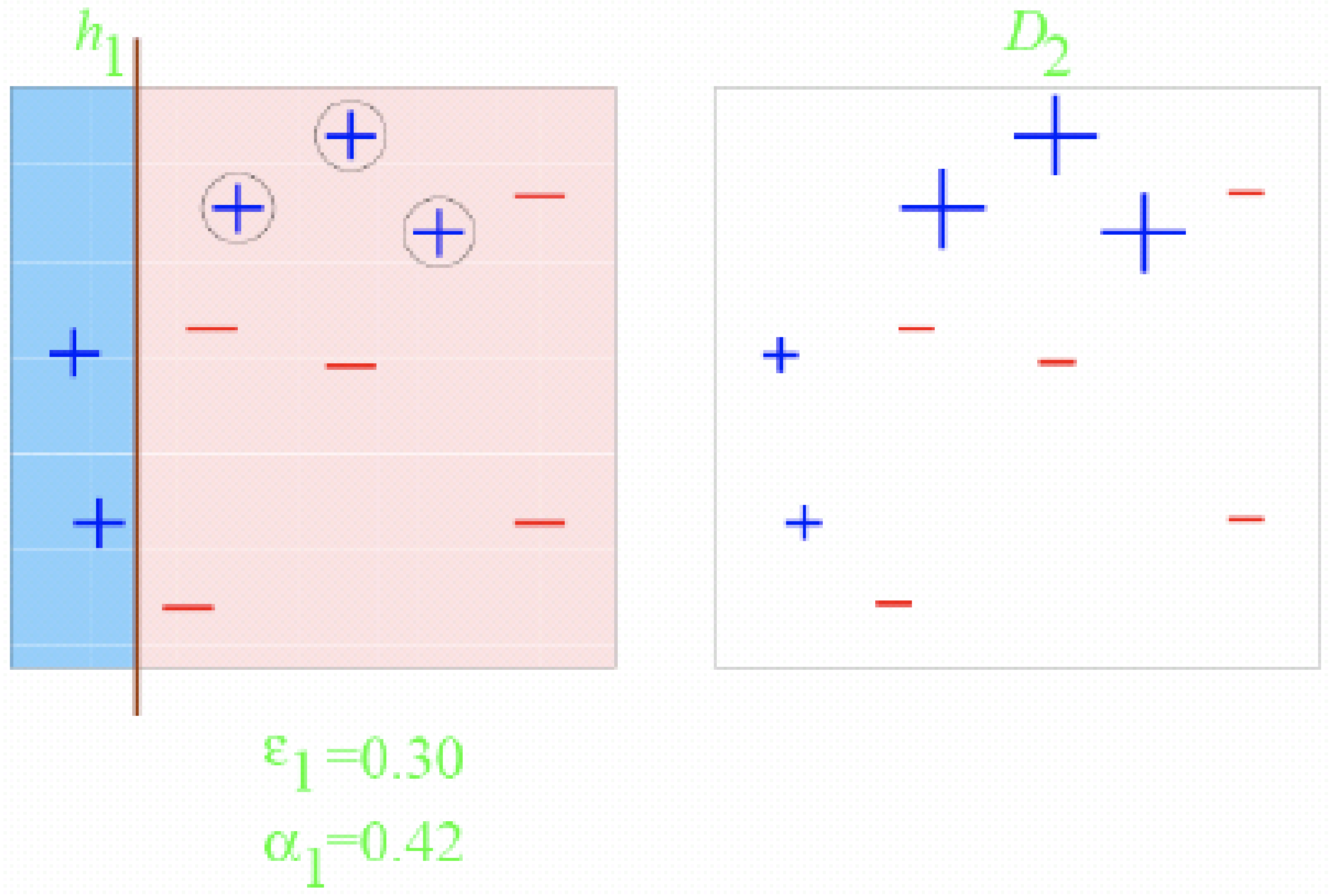
$$\alpha_i = \frac{1}{2} \ln\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

Weights:

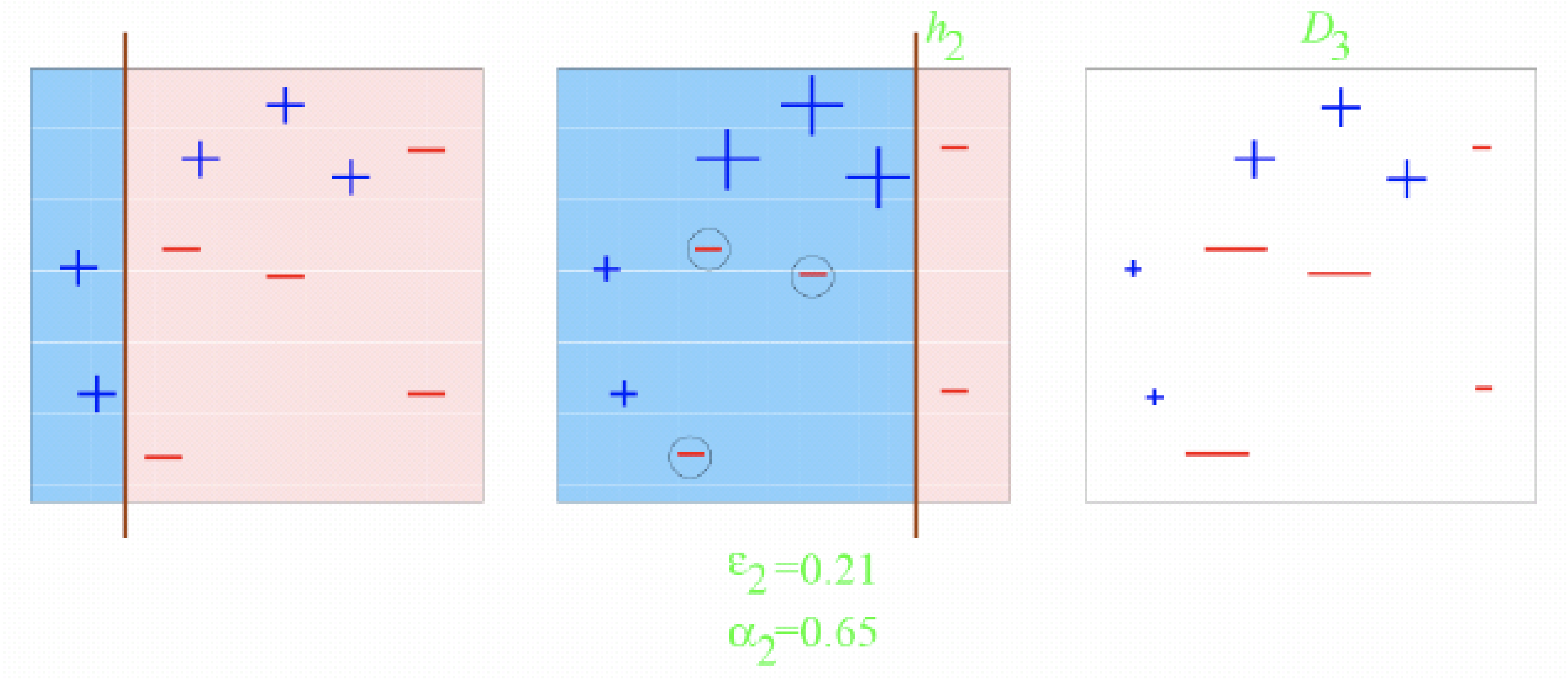
$$w_i^{(m+1)} = \frac{w_i^{(m)}}{Z_j} \begin{cases} \exp^{\alpha_j} & \text{if } y_m(x_n) \neq t_n \\ \exp^{-\alpha_j} & \text{Correct otherwise} \end{cases}$$

where Z_j is the normalization factor

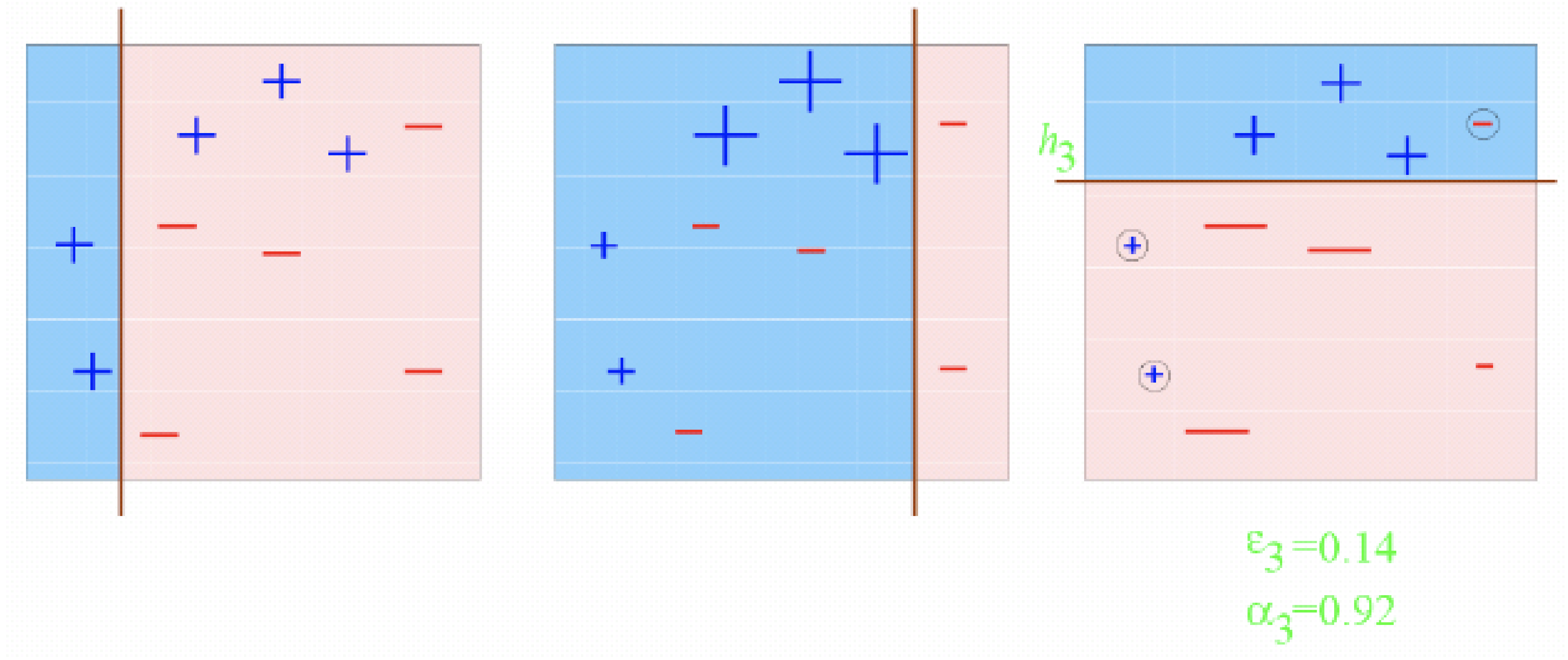
AdaBoost Example



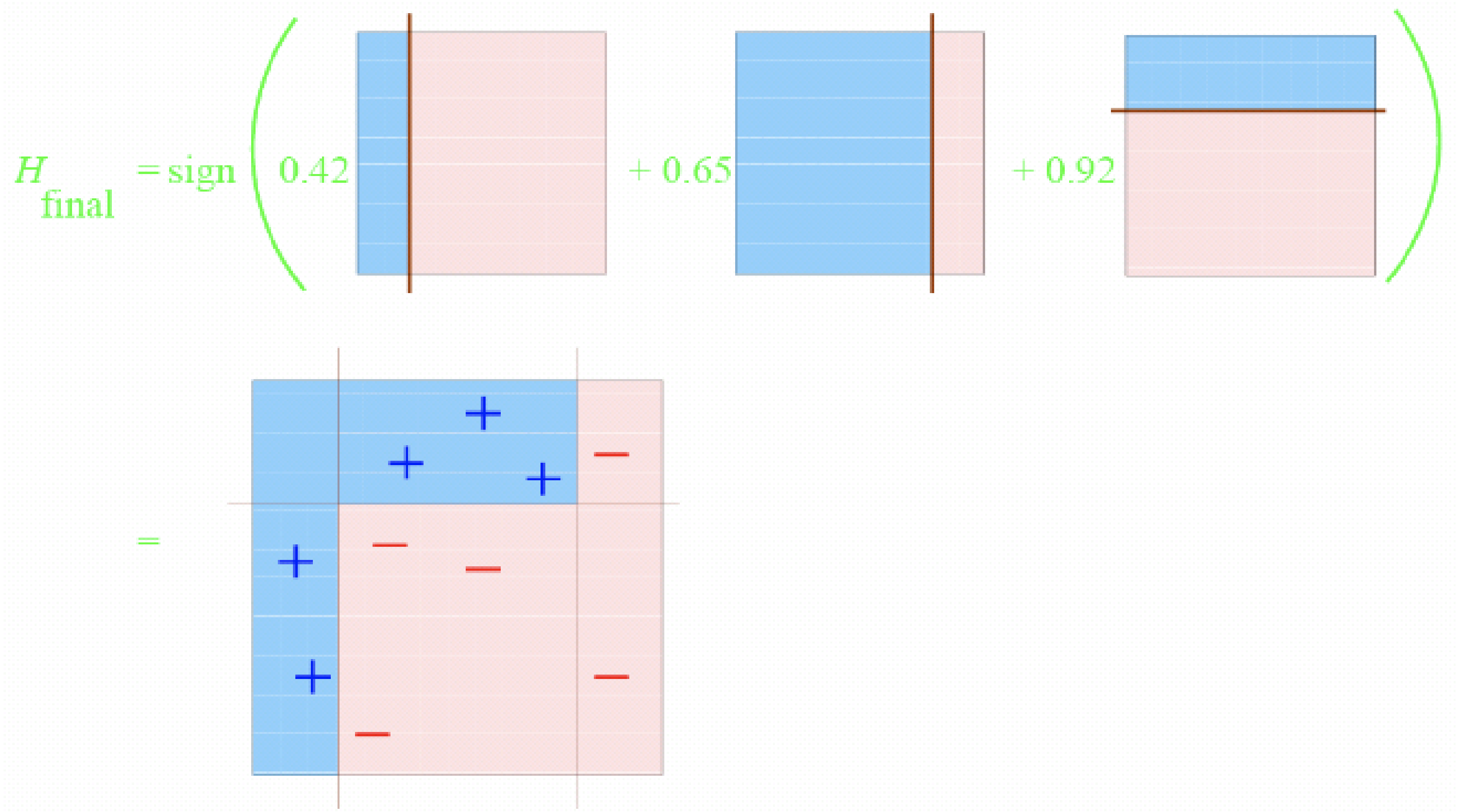
AdaBoost Example



AdaBoost Example



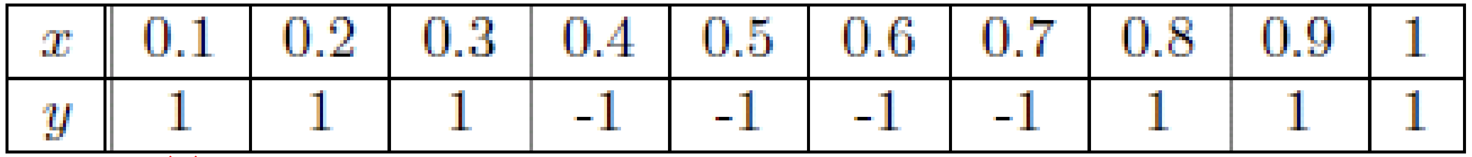
AdaBoost Example



$H_{\text{final}} = \text{sign}\left(0.42 \quad + 0.65 \quad + 0.92\right)$
=

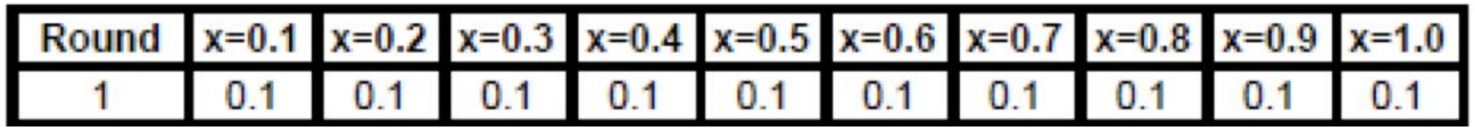
AdaBoost Example 2

Data (*x*)
Label (*y*)

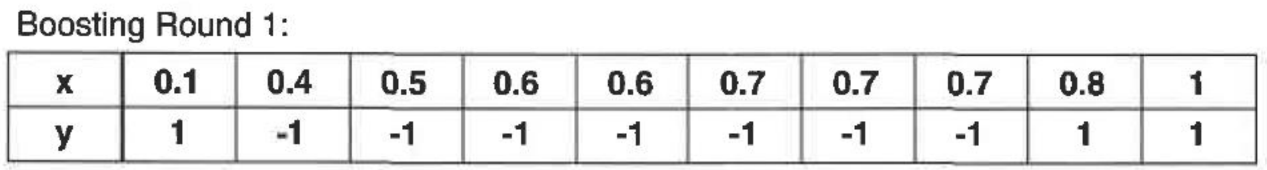


x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	-1	-1	-1	-1	1	1	1
	✖	✖	✖							

Training weights



Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1



Boosting Round 1:

x	0.1	0.4	0.5	0.6	0.6	0.7	0.7	0.7	0.8	1
y	1	-1	-1	-1	-1	-1	-1	-1	1	1

X <= 0.75 $\Longrightarrow$ -1

X > 0.75 $\Longrightarrow$ 1

$\varepsilon = (0.1 + 0.1 + 0.1) \ = 0.3$

$\alpha = 0.4236$

$e^{\alpha} = \ 1.53$

$e^{-\alpha} = 0.65$

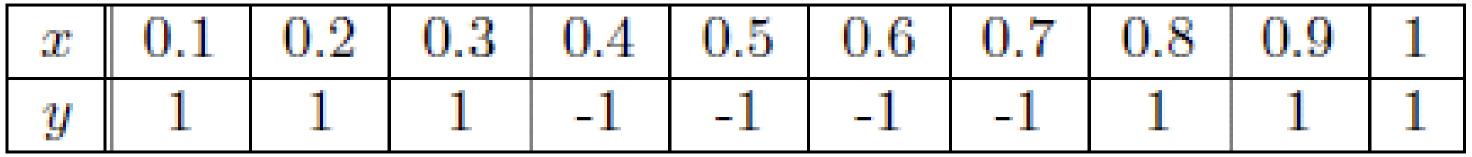
normalization $Z_1 = 3 * 0.153 + 7 * 0.065 = 0.917$

incorrect $w_1 * e^{\alpha} = \ 0.1 * 1.53 = 0.153$ $\qquad w_2 = w_1 * e^{\alpha} \,/\, Z_1 = 0.153/0.917 = 0.167$

correct $w_1 * \ e^{-\alpha} = 0.1 * 0.65 = 0.065$ $\qquad w_2 = w_1 * e^{-\alpha}/Z_1 = 0.065/0.917 = 0.071$

AdaBoost Example 2

Data (*x*)
Label (*y*)

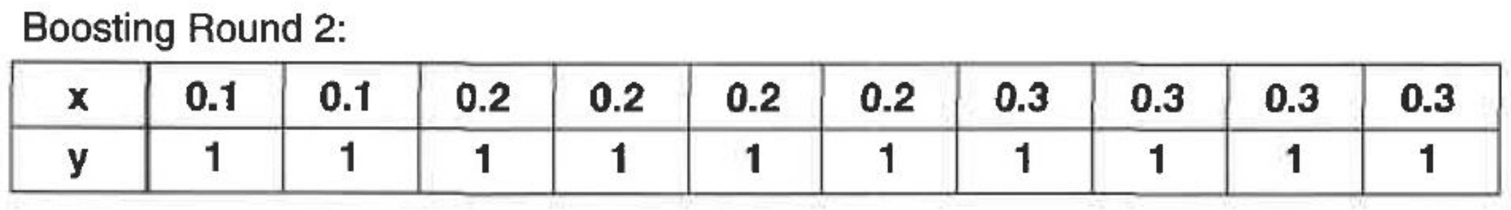


x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	-1	-1	-1	-1	1	1	1
				✖	✖	✖	✖			

weights

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0	α
1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.424
2	0.167	0.167	0.167	0.071	0.071	0.071	0.071	0.071	0.071	0.071	

sample



Boosting Round 2:

x	0.1	0.1	0.2	0.2	0.2	0.2	0.3	0.3	0.3	0.3
y	1	1	1	1	1	1	1	1	1	1

$X <= 0.05 \longrightarrow 1$

$X > 0.05 \longrightarrow 1$

AdaBoost Example 2

Data (*x*)
Label (*y*)

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	-1	-1	-1	-1	1	1	1
								✖	✖	✖

weights

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0	α
1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.424
2	0.167	0.167	0.167	0.071	0.071	0.071	0.071	0.071	0.071	0.071	0.458
3	0.12	0.12	0.12	0.13	0.13	0.13	0.13	0.05	0.05	0.05	

sample

Boosting Round 3:

x	0.2	0.2	0.4	0.4	0.4	0.4	0.5	0.6	0.6	0.7
y	1	1	-1	-1	-1	-1	-1	-1	-1	-1

$X <= 0.3 \longrightarrow 1$

$X > 0.3 \longrightarrow -1$

AdaBoost Example 2

weights

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0	α
1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.424
2	0.167	0.167	0.167	0.071	0.071	0.071	0.071	0.071	0.071	0.071	0.458
3	0.12	0.12	0.12	0.13	0.13	0.13	0.13	0.05	0.05	0.05	0.867

split

Round	Split Point	Left Class	Right Class	α
1	0.75	-1	1	0.424
2	0.05	1	1	0.458
3	0.3	1	-1	0.867

AdaBoost Example 2

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0	α
1	-1	-1	-1	-1	-1	-1	-1	1	1	1	0.424
2	1	1	1	1	1	1	1	1	1	1	0.458
3	1	1	1	-1	-1	-1	-1	-1	-1	-1	0.867

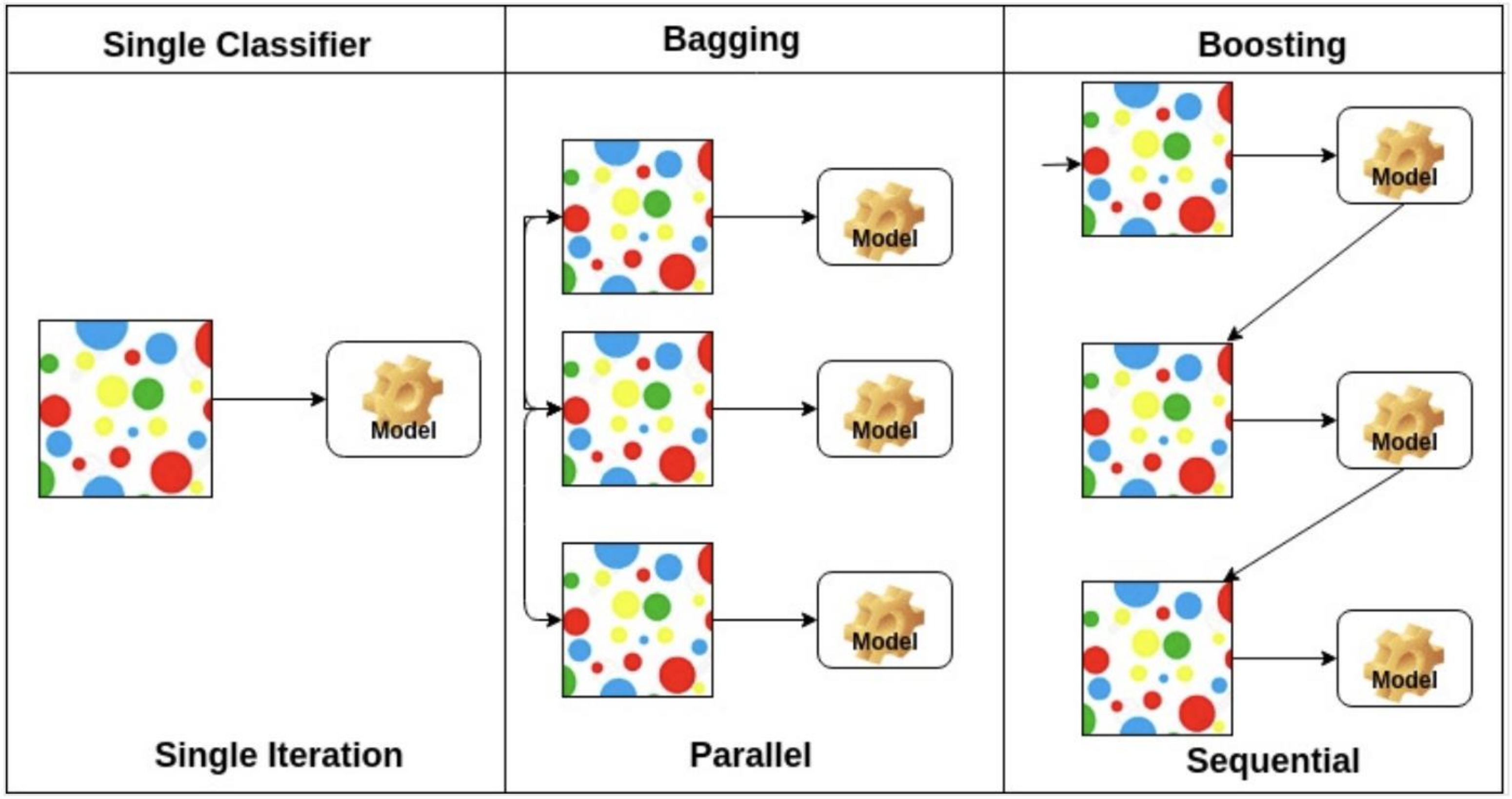
$$X(0.1) = -1 * 0.424 + 1 * 0.458 + 1 * 0.867 = 0.901 > 0, 1$$
$$X(0.9) = 1 * 0.424 + 1 * 0.458 - 1 * 0.867 = 0.015 > 0, 1$$
$$X(0.4) = -1 * 0.424 + 1 * 0.458 - 1 * 0.867 = -0.8 < 0, -1$$

AdaBoost Example 2

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0	α
1	-1	-1	-1	-1	-1	-1	-1	1	1	1	0.424
2	1	1	1	1	1	1	1	1	1	1	0.458
3	1	1	1	-1	-1	-1	-1	-1	-1	-1	0.867
Sum	0.901	0.9	0.9	-0.83	-0.83	-0.83	-0.833	0.015	0.015	0.015	
Sign	1	1	1	-1	-1	-1	-1	1	1	1	
Ground	1	1	1	-1	-1	-1	-1	1	1	1	

Round	Split Point	Left Class	Right Class	α
1	0.75	-1	1	0.424
2	0.05	1	1	0.458
3	0.3	1	-1	0.867

Comparison


Single Classifier
Bagging
Boosting
Model
Model
Model
Model
Model
Model
Model
Single Iteration
Parallel
Sequential

Summary

- Ensemble methods work better with unstable classifiers. i.e.. Base classifiers that are sensitive to minor perturbations in the training set.
- Examples of unstable classifiers include decision trees, rule-based classifiers, and artificial neural networks.

References

“Introduction to Data Science" by Pang-Ning Tan, Michael Steinbach and Vipin Kumar