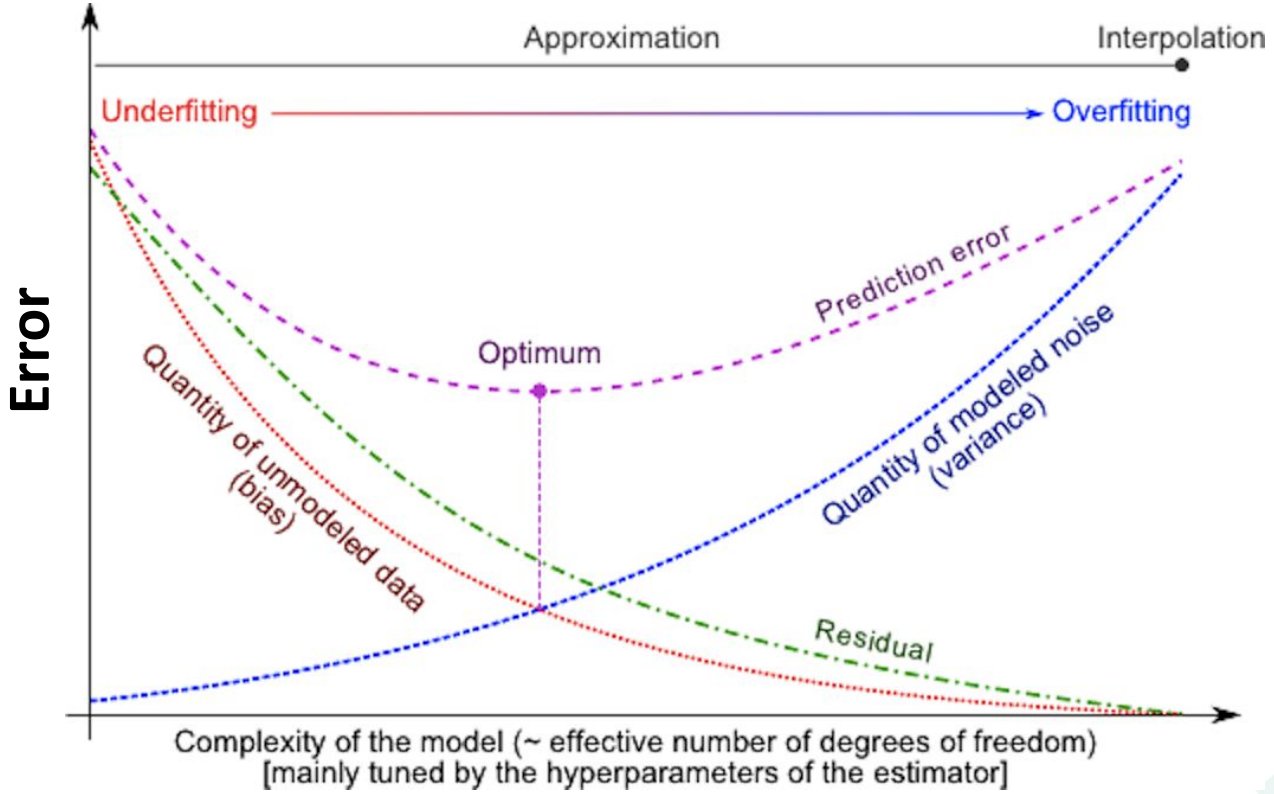
Random Forests

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY
DELHI

Bias vs. Variance Analysis



Reduce Variance Without Increasing Bias

- Averaging reduces variance:

$$Var(\overline{X}) = \frac{Var(X)}{N}$$

- Average models to reduce model variance
- Problem: Only one training set
  - Where do multiple models come from?
- Ensemble learning
  - Combining *weak classifiers* (of the same type) in order to produce a strong classifier
  - Condition: diversity among the weak classifiers
- *Weak classifiers*: only need to be better than random guess

Decision Trees -> Random Forests

- DT are non-flexible -> Inaccurate
  - Performance on unseen data suffers -> High Variance
- Definition: Random Forests
  - Collection of unpruned CARTs
  - Rule to combine individual tree decisions
- Ensemble learning: Two ways to introduce randomness/diversity
  - “Bagging” and “Random input vectors”
    - Bagging method: each tree is grown using a bootstrap sample of training data
    - Random vector method: At each node, best split is chosen from a random sample of m attributes instead of all attributes.

Random Forests

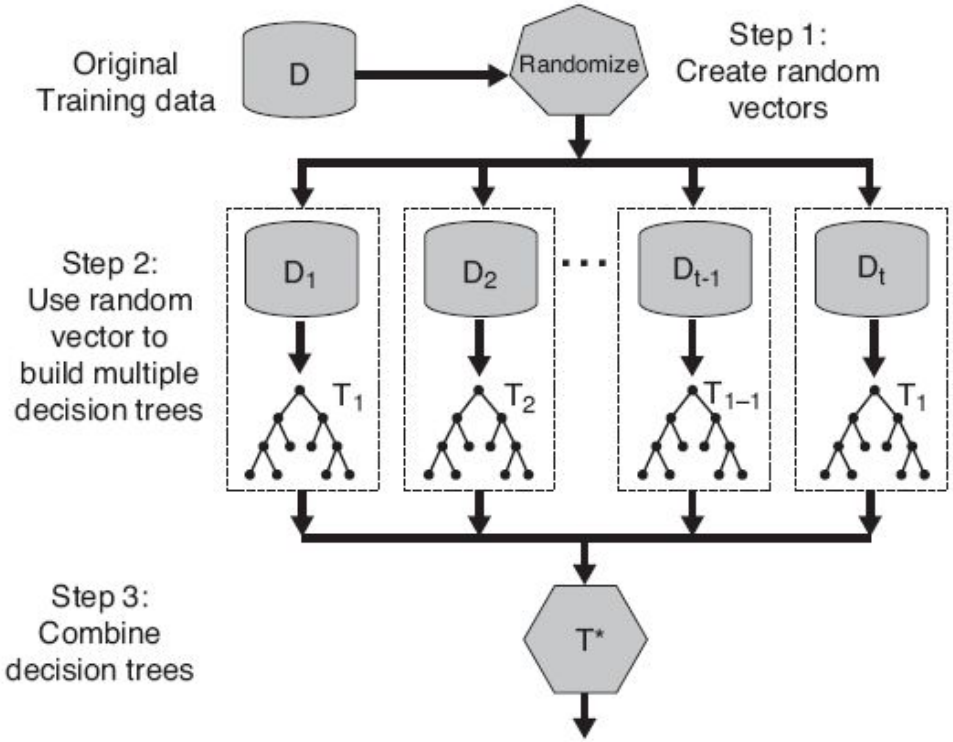


Figure 5.40. Random forests.

Bagging

- L: original learning set composed of n samples
- Generate K learning sets L_k...
  - ... composed of m samples, $m \leq n$,...
  - ... obtained by uniform sampling with replacement from L
  - In consequences, L_k may contain repeated samples

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

0.632 Bootstrap

- $m = n$
- A particular training data has a probability of (*1-1/n*) of not being picked
- Thus its probability of ending up in the test data (not selected) is, when picking *n* data points :

$$\left(1-\frac{1}{n}\right)^{n} \approx e^{-1} = 0.368$$

- -> Training data will contain approximately 63.2% of the instances
- Out of Bag (OOB) samples-> Out-Of-Bag Error

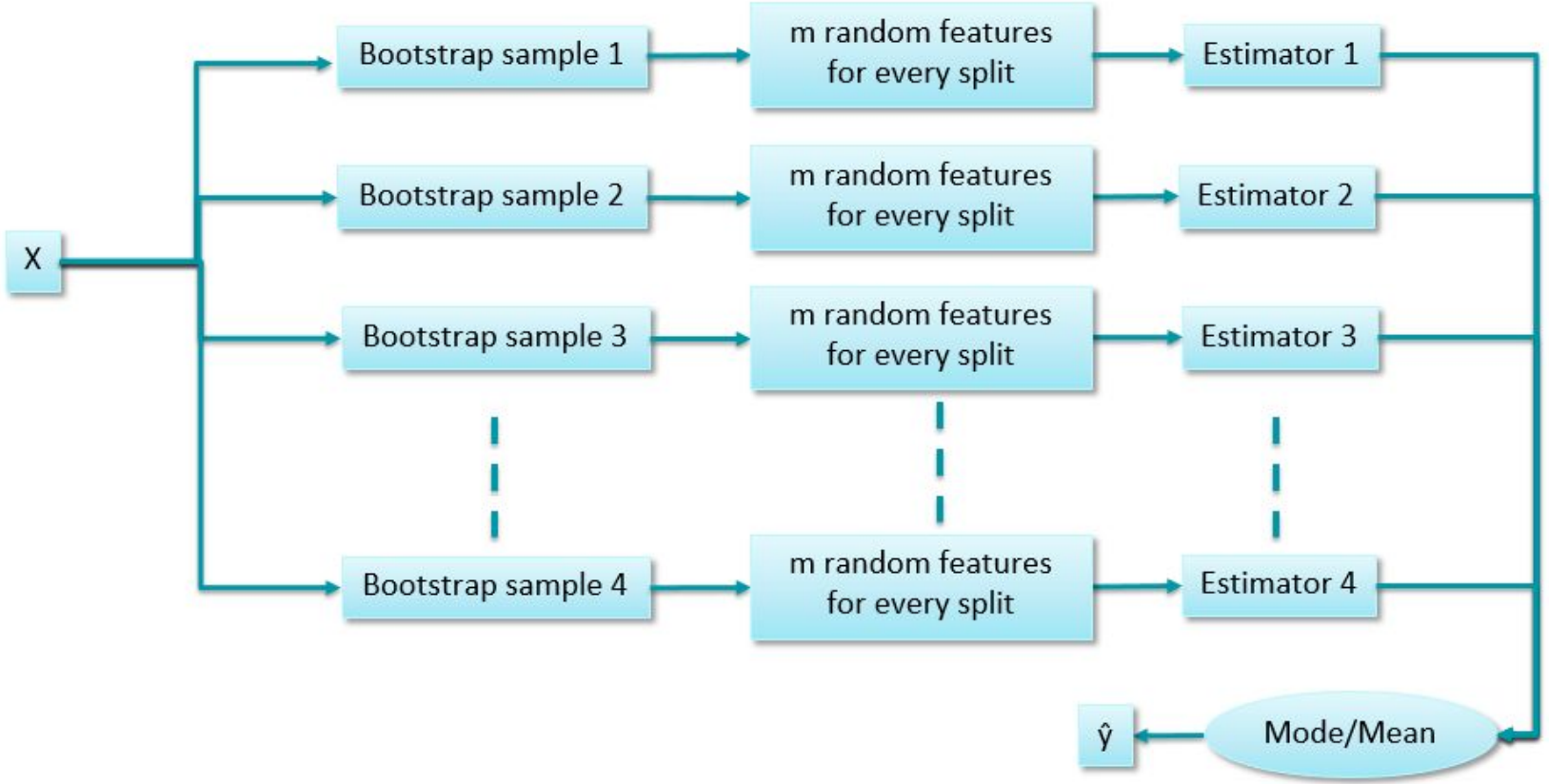
Out-of-bag error

- For each sample *S* of the OOB set
  - Predict the class of S with OOB classifiers that does not contain the sample *S*
  - Error = is prediction correct?
- Out-of-bag error = average over all samples of *S*
- Provides an estimation of the generalization error
  - Can be used to decide when to stop adding trees to the forest
  - As accurate as using a test set of the same size as the training set.
    - Therefore, using the out-of-bag error estimate removes the need for a set aside test set.

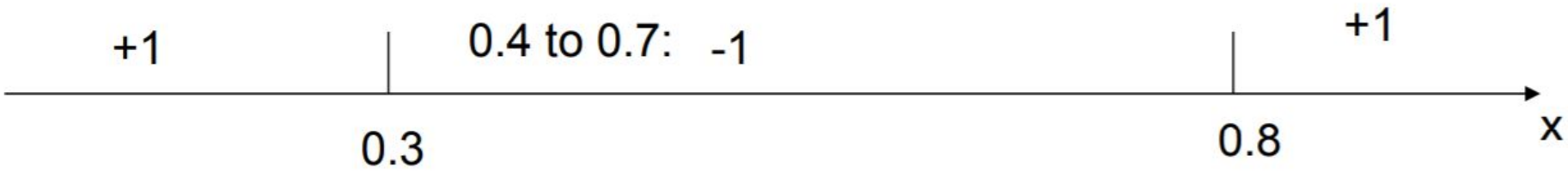
Random forest > Bagging > Aggregation

- Bagging: Bootstrap aggregation
- Technique of ensemble learning
  - To avoid over-fitting
    - Important since trees are unpruned
  - To improve stability and accuracy
- Idea
  - Create N bootstrap samples $\{S_1, \dots, S_N\}$ of S as follows:
    - For each S_i randomly draw $|S|$ examples from S with replacement
  - For each $i = 1, \dots, N$, $h_i = \text{Learn}(S_i)$
  - Output $H = <\{h_1, \dots, h_N\}, majorityVote>$
    - Use average or majority voting to aggregate results

Random forest > Bagging > Aggregation



Bagging Example -1



Goal: Find a collection of 10 simple thresholding classifiers that collectively can classify correctly.

- Each simple (or weak) classifier is:

  (x<=K class = +1 or -1 depending on which value yields the lowest error)

Bagging Example - I

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 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 Example - II

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 Example - III

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 Example - IV

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

Method for Growing the tree

- Fix a $m <= M$. At each node
- Method 1
  - Choose m attributes randomly, compute their information gains, and choose the attribute with the largest gain to split
- Method 2: (When M is not very large):
  - Select L of the attributes randomly. Compute a linear combination of the L attributes using weights generated from [-1,+1] randomly. That is, new A = Sum(W_i*M_i), i=1..L.

Method for Growing the tree

- Method 3:
  - Compute the information gain of all M attributes. Select the top m attributes by information gain. Randomly select one of the m attributes as the splitting node.

Value of m

- For classification, the default value for m is $\lfloor\sqrt{M}\rfloor$ and the minimum node size is one.
- For regression, the default value for m is $\lfloor M/3 \rfloor$ and the minimum node size is five.

Reducing Variance

- The variance of B i.i.d. random variables $\frac{\sigma^2}{B}$
  - B = number of trees, σ^2 = variance of each individual tree

- If the variables are simply i.d. (identically distributed, but not necessarily independent) with positive pairwise correlation ρ the variance of the ensemble is $\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$

- The variance of an ensemble is strictly smaller than the variance of an individual model.

Reducing Variance

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

- As B increases, the second term disappears
  - Size of the correlation of pairs of bagged trees limits the benefits of averaging
- To improve the variance reduction of bagging by reducing the correlation ρ between the trees, without increasing the variance too much
  - Random selection of the input variables during tree growing
  - Specifically, before each split, select $m<=p$ of the input variables at random as candidates for splitting

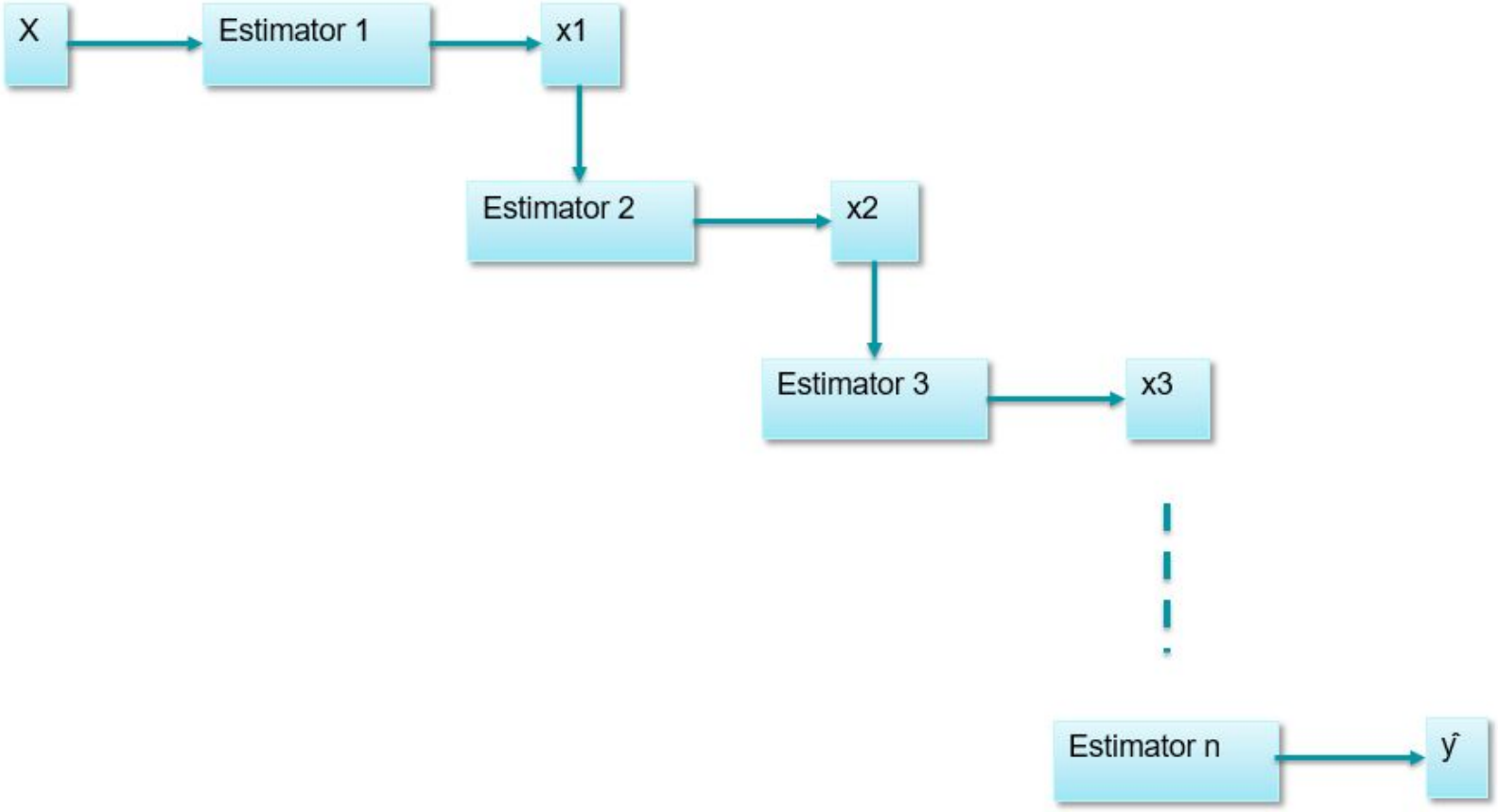
Reducing Bias and Variance

- Bagging: Each individual classifier is independent
  - Bagging has little effect on bias
- Each tree generated in bagging is identically distributed (i.d.), the expectation of an average of B such trees is
  - the same as the expectation of any one of them
    - The bias of bagged trees is the same as that of the individual trees
- Can we average and reduce bias?
  - YES
    - Boosting
- In Boosting the trees are grown in an adaptive way to remove bias, and hence are not i.d.

Boosting

- Boosting is iterative and adaptive:
  - Looks at the errors from previous classifiers to decide what to focus on for the next iteration
  - Successive classifiers depend on their predecessors
  - Key idea: place more weight on “hard” examples (i.e., instances that were misclassified on previous iterations)
  - Records that are classified correctly will have their weights decreased
- Adaboost – popular boosting algorithm

Boosting



Bagging and Boosting Summary

Bagging

- Resample data points
- Weight of each classifier is the same
- Only variance reduction
- Robust to noise and outliers

Boosting

- Reweight data points(modify data distribution)
- Weight of classifier vary depending on accuracy
- Reduces both bias and variance
- Can hurt performance with noise and outliers

References

1. The Elements of Statistical Learning, Chapter 15
2. Efron, B. and R. Tibshirani (1997), "Improvements on Cross-Validation: The .632+ Bootstrap Method," Journal of the American Statistical Association Vol. 92, No. 438. (Jun), pp. 548-560
3. https://perso.math.univ-toulouse.fr/motimo/files/2013/07/random-forest.pdf
4. https://stat.ethz.ch/education/semesters/ss2012/ams/slides/v10.2.pdf
5. StatQuest: Random Forests Part 1 - Building, Using and Evaluating
    a. Look at Part 2 also.