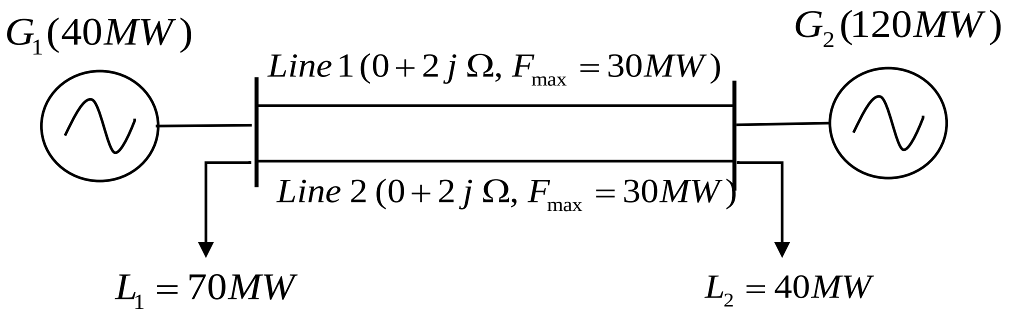
Big Data Technology and its Applications

Ensemble learning and Random forest

张宁 ningzhang@tsinghua.edu.cn

A problem in power system



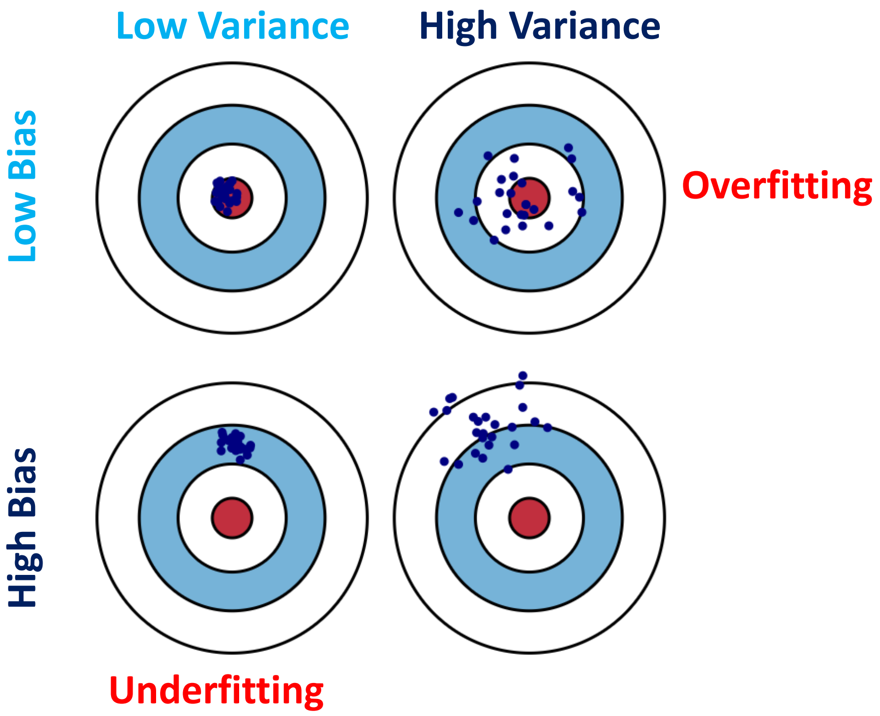
- Given a set of operation state and assuming loads are constant, how to judge whether the power system is safe?

ID	G1 generation	G2 generation	Line 1 status	Safe or not
1	0	110	Connected	**N**
2	20	90	Connected	**Y**
3	40	70	Connected	**Y**
4	0	110	Disconnected	**N**
5	20	90	Disconnected	**N**
6	40	70	Disconnected	**Y**

- Lots of similar problems in power systems.

Bias and Variance

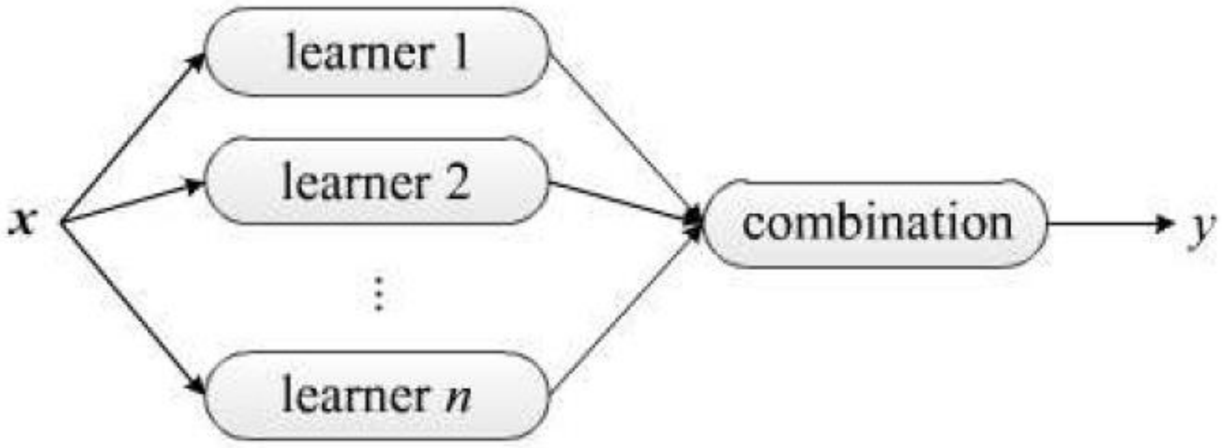
- **Bias:** the error due to bias is taken as the difference between the expected (or average) prediction of our model and the correct value which we are trying to predict.
- **Variance:** the error due to variance is taken as the variability of a model prediction for a given data point.
- **Generalization error** = Bias^2 + Variance + Noise



Ref: http://scott.fortmann-roe.com/docs/BiasVariance.html

Ensemble learning

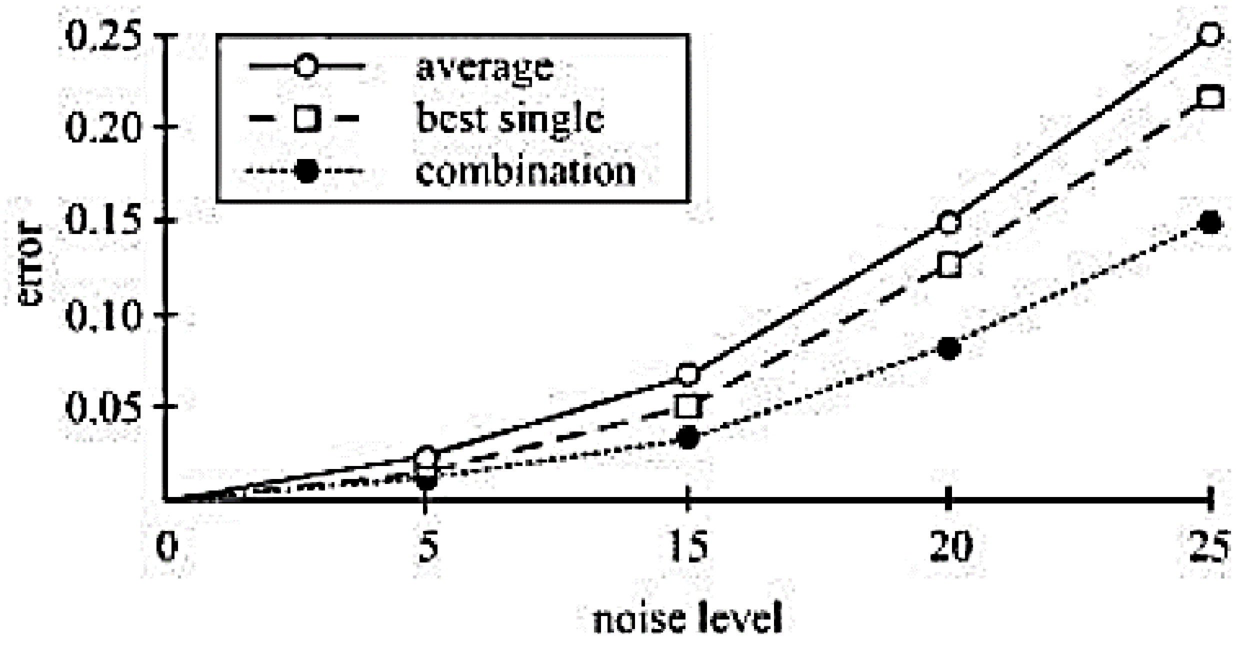
- Ensemble learning trains multiple base learners to solve same problem
- The base learner also called weak learner can be any learning algorithm like decision tree or neural network
- Why ensemble (昂桑宝)?
- The generalization ability of an ensemble is often stronger than that of base learners



From Zhihua Zhou, Ensemble Methods: Foundations and Algorithms, 2012

Ensemble learning foundation

- Hansen and Salamon (1990) found that predictions made by the combination of a set of classifiers are often more accurate than predictions made by the best single classifier.
- Schapire (1990) proved that weak learners can be boosted to strong learners



From Hansen and Salamon, 1990

Ensemble learning application

- Ensemble method like random forest and xgboost are widely used in machine learning and data driven challenges
- Random forests have lead to one of the biggest success stories of computer vision on the Microsoft Kinect for XBox 360 in 2011
- Ensemble learning (Xgboost) was used by 17 solutions among the 29 challenge winning solutions published at Kaggle’s blog during 2015
- Ensemble learning (Xgboost) was used by every winning team in the top-10 of KDDCup 2015 （国际知识发现和数据挖掘竞赛）

Construct a good ensemble

- Generating base learner, like decision tree
- Combining the base learner
- Ensemble principle (accurate and diverse)
  - The base learner should be as accurate as possible 好
  - As diverse as possible 而不同

Ensembles: Parallel vs Sequential

- Parallel ensembles: each model is built independently
  - e.g. bagging and random forests
  - Main Idea: Combine many (high complexity, low bias) models to reduce variance
- Sequential ensembles:
  - Models are generated sequentially
  - Try to add new models that do well where previous models lack

Bagging

- We want to get base learners as independent as possible.
- However, sampling a number of non-overlapped data subsets will produce very small and unrepresentative samples, leading to poor performance of base learners.
- Bagging (Bootstrap AGGregatING) uses bootstrap and aggregation.
- Bagging adopts averaging for regression and voting for classification.

Benefit of averaging

- Let $z, z_1, \ldots, z_n$ be *i.i.d.* with $\mathbb{E}z = \mu$ and $\mathrm{Var}(z) = \sigma^2$
- Average has the same expected value but smaller standard error:

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} z_i\right] = \mu \quad \mathrm{Var}\left[\frac{1}{n}\sum_{i=1}^{n} z_i\right] = \frac{\sigma^2}{n}$$

- If the $z, z_1, \ldots, z_n$ represent estimators trained with independent training samples from same distribution, clearly the average is preferred to a single estimator.
- How to get the independent training samples?
- Bootstrap (自助法)!

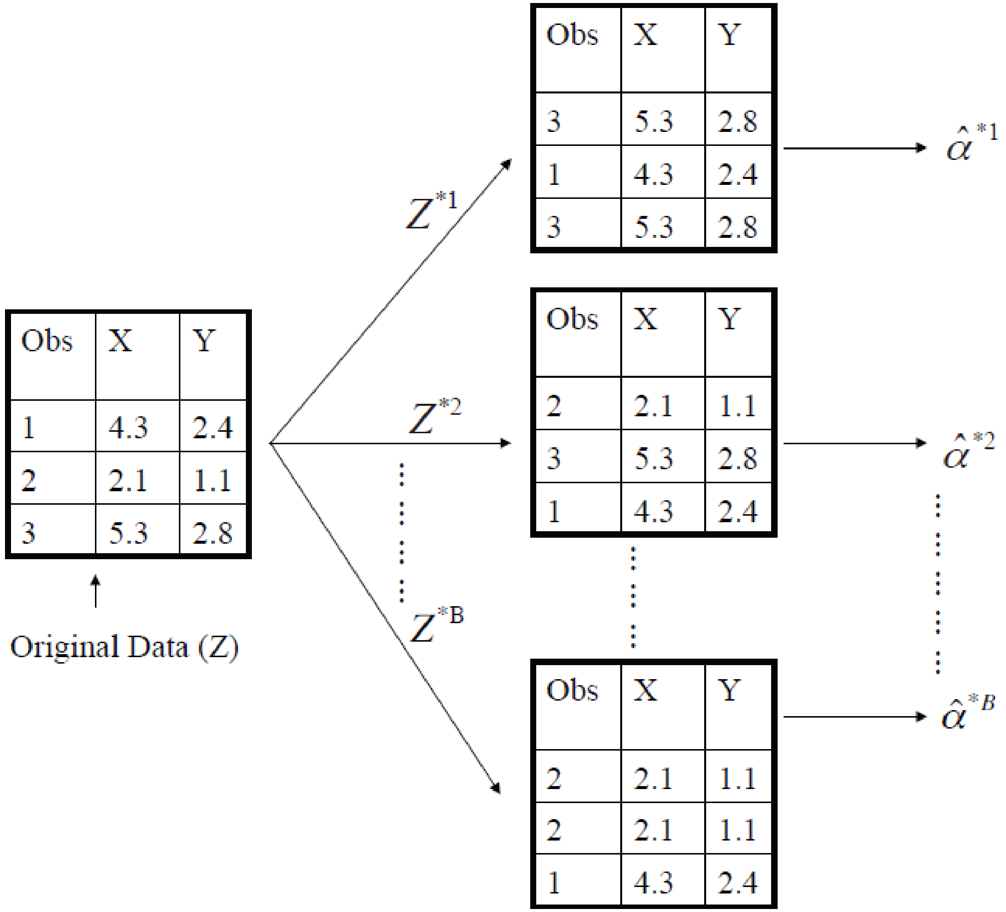
Bootstrap sample

- A bootstrap sample from $\mathcal{D}_n = x_1, \ldots, x_n$ is a sample of size *n* drawn with replacement from $\mathcal{D}_n$.
- In a bootstrap sample, some elements will show up multiple times, and some won't show up at all.
- Each instance in dataset has a following probability of not being selected.

$$\left(1 - \frac{1}{n}\right)^n \approx \frac{1}{e} \approx .368 \quad \text{if } n \to \infty$$

- So we expect ~63.2% of elements of dataset $\mathcal{D}_n$ will show up at least once

Bootstrap sample



From An Introduction to Statistical Learning, with applications in R (Springer, 2013)

Averaging combination methods

- Given a set of individual learners $h_1, \ldots, h_T$, and the output of h_i for the instance x is $h_i(x)$
- Simple averaging

$$H(\mathbf{x}) = \frac{1}{T} \sum_{i=1}^{T} h_i(\mathbf{x})$$

- Weighted averaging

$$H(\mathbf{x}) = \sum_{i=1}^{T} w_i h_i(\mathbf{x}) \quad \text{where } w_i \geq 0 \text{ and } \sum_{i=1}^{T} w_i = 1$$

- In general, simple averaging is appropriate for combining learners with similar performances
- Whereas if the individual learners exhibit nonidentical strength, weighted averaging with unequal weights may achieve a better performance.

Voting combination methods

- For classification task with class label $c_1, \dots, c_l$
- $h_i^j(x)$ is output of h_i for class c_j
- Majority voting

$$H(\mathbf{x}) = \begin{cases} c_j & \text{if } \sum_{i=1}^{T} h_i^j(\mathbf{x}) > \frac{1}{2} \sum_{k=1}^{l} \sum_{i=1}^{T} h_i^k(\mathbf{x}) \\ \text{rejection} & \text{otherwise} \end{cases}$$

- Plurality voting

$$H(\mathbf{x}) = c\{\arg\max_j \sum_{i=1}^{T} h_i^j(\mathbf{x})\}$$

- Weighted Voting

$$H(\mathbf{x}) = c\{\arg\max_j \sum_{i=1}^{T} w_i h_i^j(\mathbf{x})\} \quad \text{where } w_i \geq 0 \text{ and } \sum_{i=1}^{T} w_i = 1$$

- Hard voting and soft voting $h_i^j(x) \in \ 0,1$ or $[0,1]$

Bagging algorithm

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

Process:
for $t = 1, \cdots, T$:
$\mathcal{D}_t = Bootstrap(\mathcal{D})$; % Generate a bootstrap sample from $\mathcal{D}$
$h_t = \mathfrak{L}(\mathcal{D}_t)$ % Train a base learner h_t from the bootstrap sample
end.

Output: $H(\boldsymbol{x}) = \text{argmax}_{y \in \mathcal{Y}} \sum_{t=1}^{T} \mathbb{1}(y = h_t(\boldsymbol{x}))$ % the value of $\mathbb{1}(a)$ is 1 if a is $true$ and 0 otherwise

From Zhihua Zhou, Ensemble Methods: Foundations and Algorithms, 2012

Out of bag estimation

- Each bagged predictor is trained on about 63% of the data
- Remaining 37% are called out-of-bag (OOB) observations
- The OOB error is a good estimate of the generalization error of base learner.

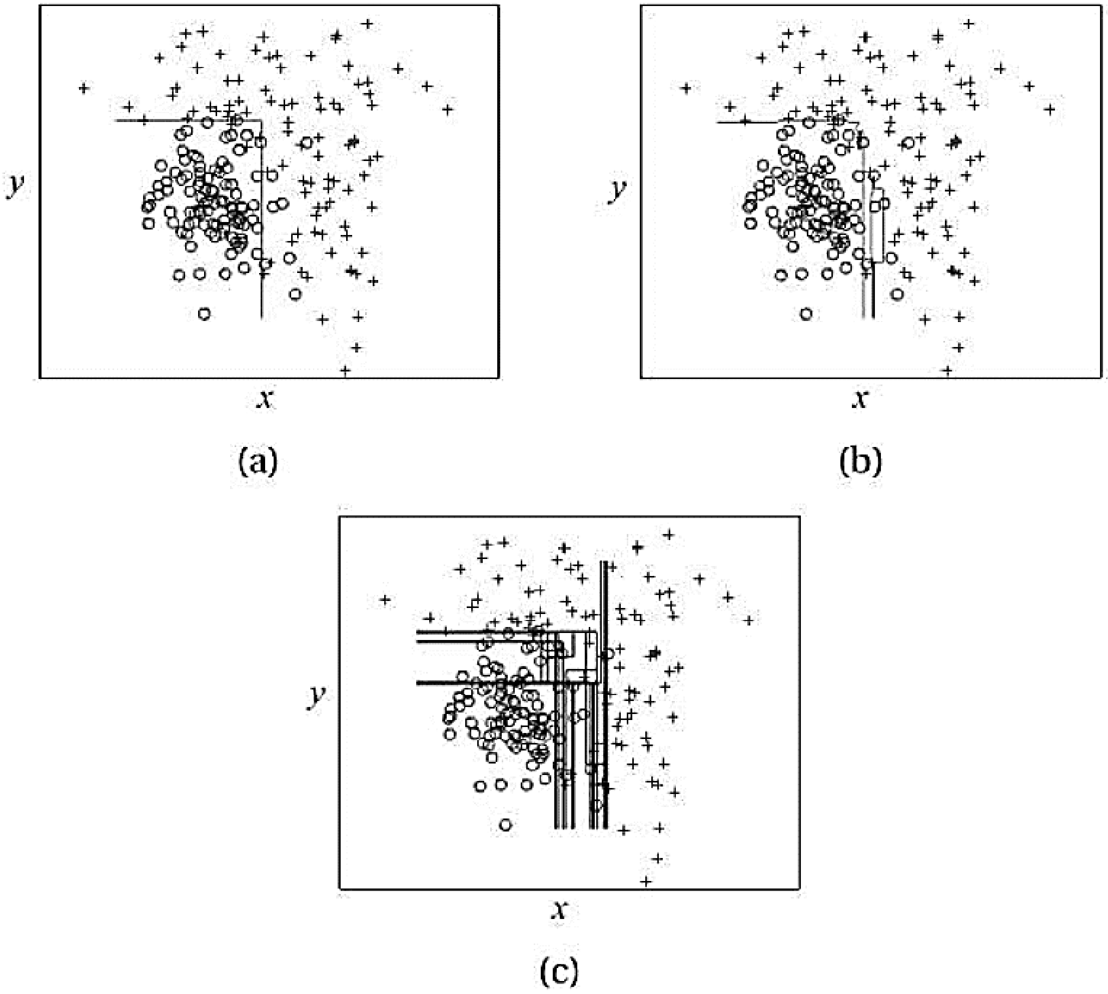
- OOB prediction

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

- OOB error

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

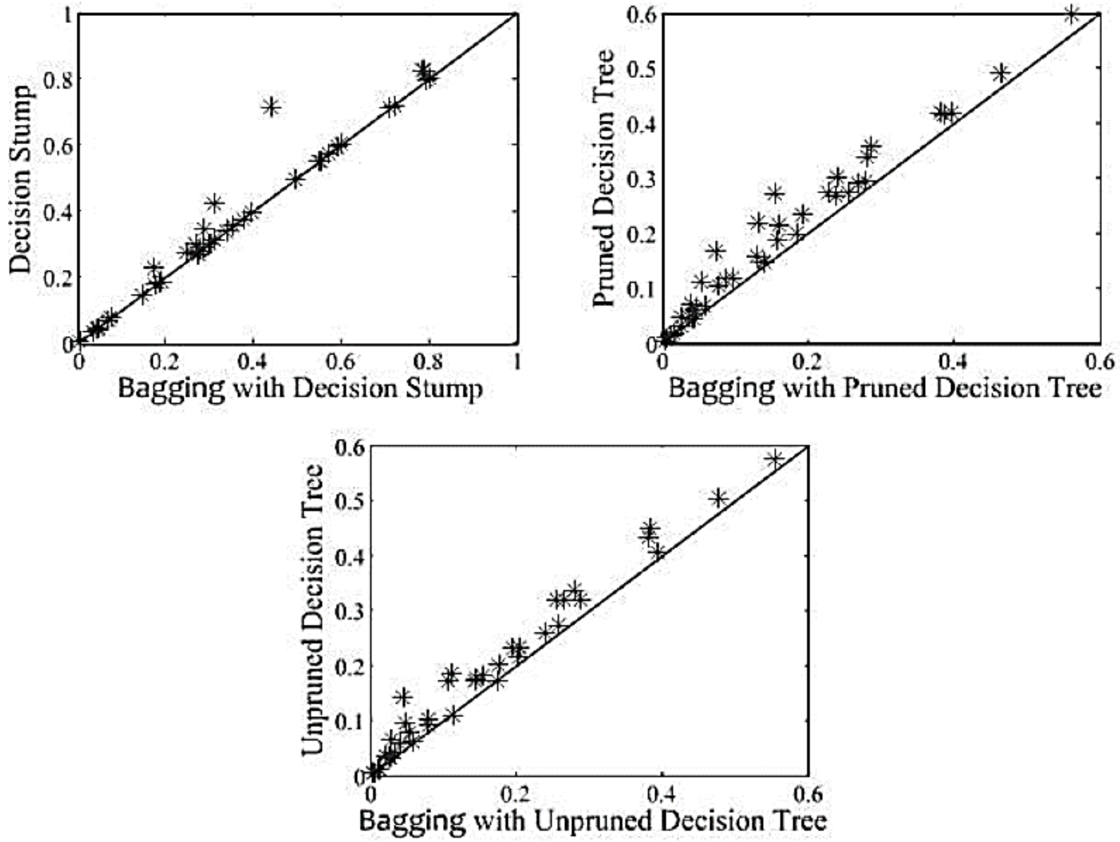
Bagging examples



Decision boundaries of (a) a single decision tree, (b) Bagging and (c) the 10 decision trees used by Bagging, on the three-Gaussians data set.

From Zhihua Zhou, Ensemble Methods: Foundations and Algorithms, 2012

Bagging examples



Comparison of predictive errors of Bagging against single base learners on 40 UCI data sets. Each point represents a data set and locates according to the predictive error of the two compared algorithms. The diagonal line indicates where the two compared algorithms have identical errors.

From Zhihua Zhou, Ensemble Methods: Foundations and Algorithms, 2012

Bagging application tips

- Bagging reduces variance without making bias worse.
- General sentiment is that bagging helps most when
  - Relatively unbiased base prediction functions
  - High variance / low stability
  - i.e. small changes in training set can cause large changes in predictions

Limitation of bagging

- Averaging estimators reduces variance if they’re based on *i.i.d.* samples from real distribution
- Bootstrap samples are
  - independent samples from the training set, but are not independent samples from real distribution.
- This dependence limits the amount of variance reduction we can get

Variance of a Mean of Correlated Variables

- For $Z, Z_1, \dots, Z_n$ i.i.d. with $\mathbb{E}Z = \mu$ and $\operatorname{Var} Z = \sigma^2$,

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} Z_i\right] = \mu \qquad \operatorname{Var}\left[\frac{1}{n}\sum_{i=1}^{n} Z_i\right] = \frac{\sigma^2}{n}.$$

- What if Z's are correlated?
- Suppose $\forall i \neq j$, $\operatorname{Corr}(Z_i, Z_j) = \rho$. Then

$$\operatorname{Var}\left[\frac{1}{n}\sum_{i=1}^{n} Z_i\right] = \rho\sigma^2 + \frac{1-\rho}{n}\sigma^2.$$

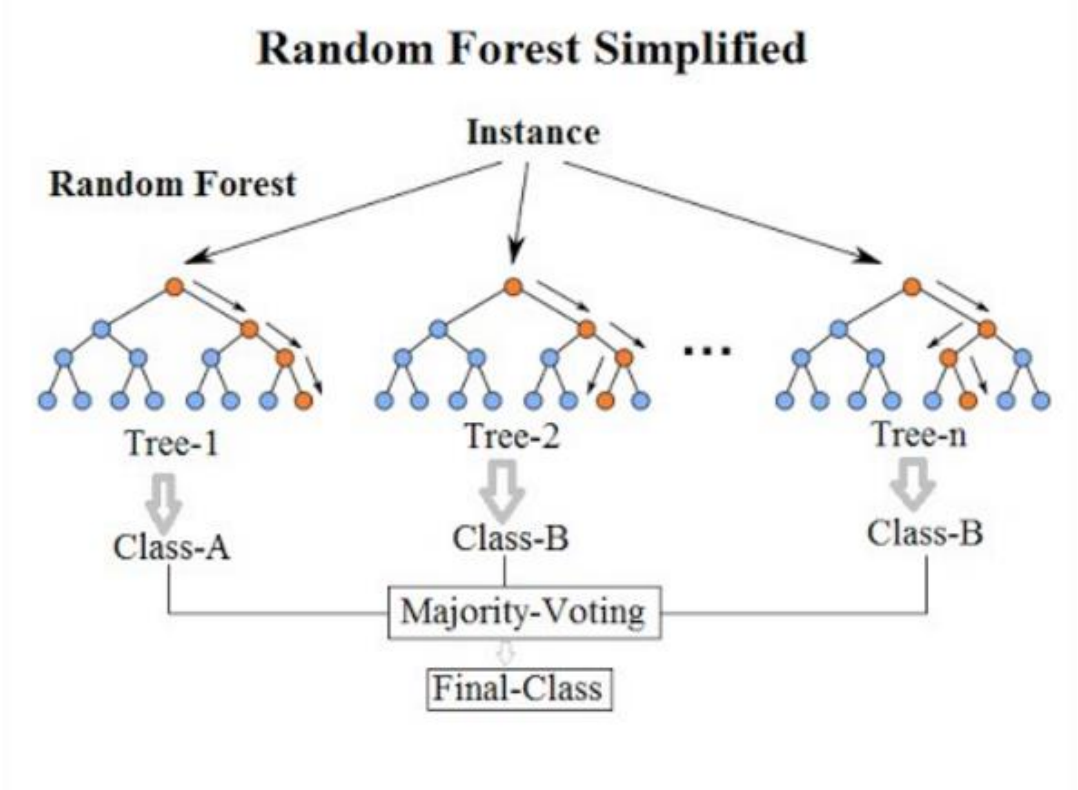
- For large n, the $\rho\sigma^2$ term dominates – limits benefit of averaging.

From David S. Rosenberg,DS-GA 1003

Random forest

Main idea

- Use bagged decision trees, but modify the tree-growing procedure to reduce the dependence between trees.
- Random select features for individual tree



Key step in random forests:

- When constructing each tree node, restrict choice of splitting variable to a randomly chosen subset of features of size m.
- Typically choose

$$m = \log_2 p \text{ or m} = \sqrt{p}$$

- where p is the number of features.
- Can choose depth using cross validation.

Random forests algorithm

Input: Data set $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
Feature subset size K.

Process:

1. $N \leftarrow$ create a tree node based on D;
2. **if** *all instances in the same class* **then return** N
3. $\mathcal{F} \leftarrow$ the set of features that can be split further;
4. **if** $\mathcal{F}$ *is empty* **then return** N
5. $\tilde{\mathcal{F}} \leftarrow$ select K features from $\mathcal{F}$ randomly;
6. $N.f \leftarrow$ the feature which has the best split point in $\tilde{\mathcal{F}}$;
7. $N.p \leftarrow$ the best split point on $N.f$;
8. $D_l \leftarrow$ subset of D with values on $N.f$ smaller than $N.p$;
9. $D_r \leftarrow$ subset of D with values on $N.f$ no smaller than $N.p$;
10. $N_l \leftarrow$ call the process with parameters (D_l, K);
11. $N_r \leftarrow$ call the process with parameters (D_r, K);
12. **return** N

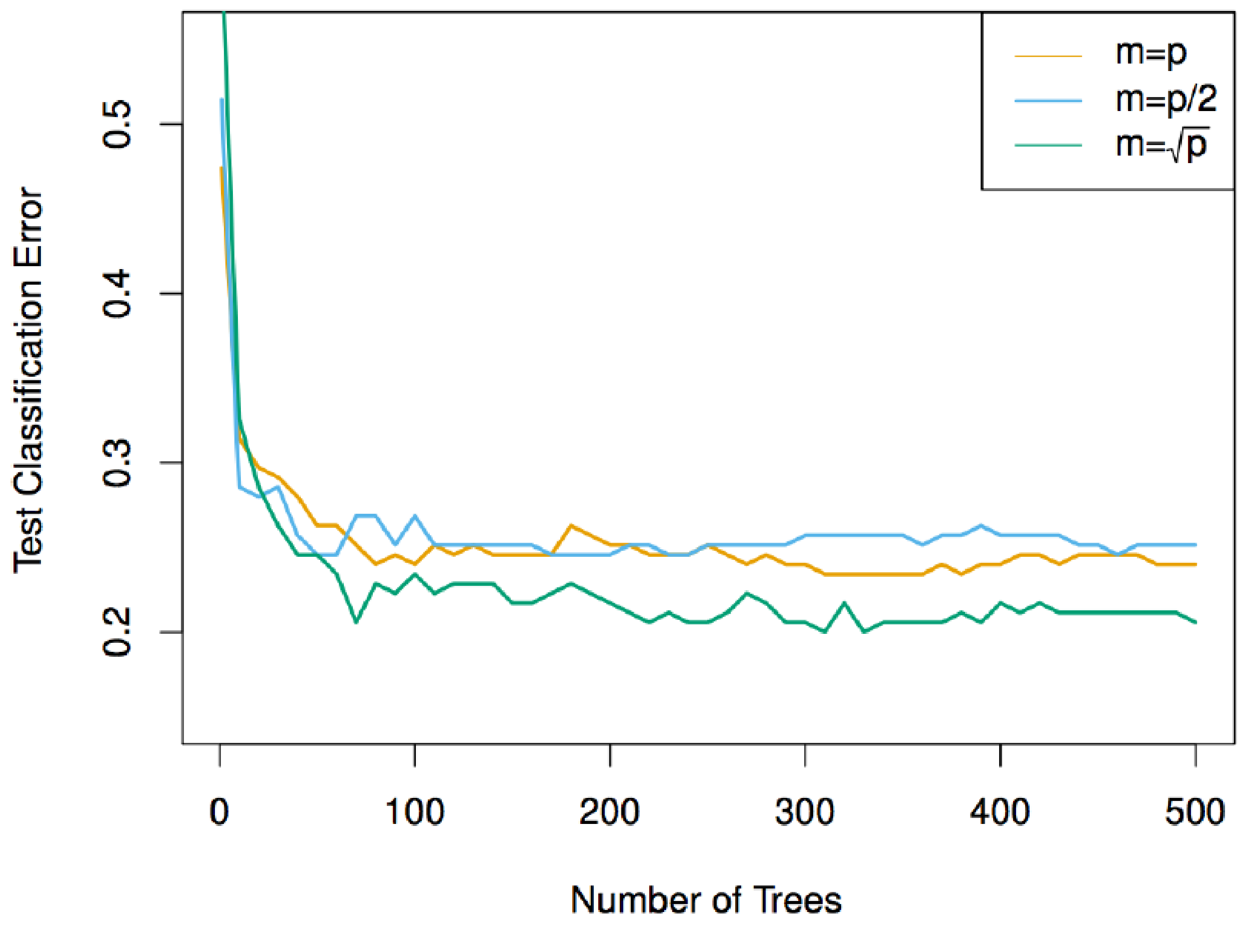
Output: A random decision tree

From Zhihua Zhou, Ensemble Methods: Foundations and Algorithms, 2012

Random forest tips

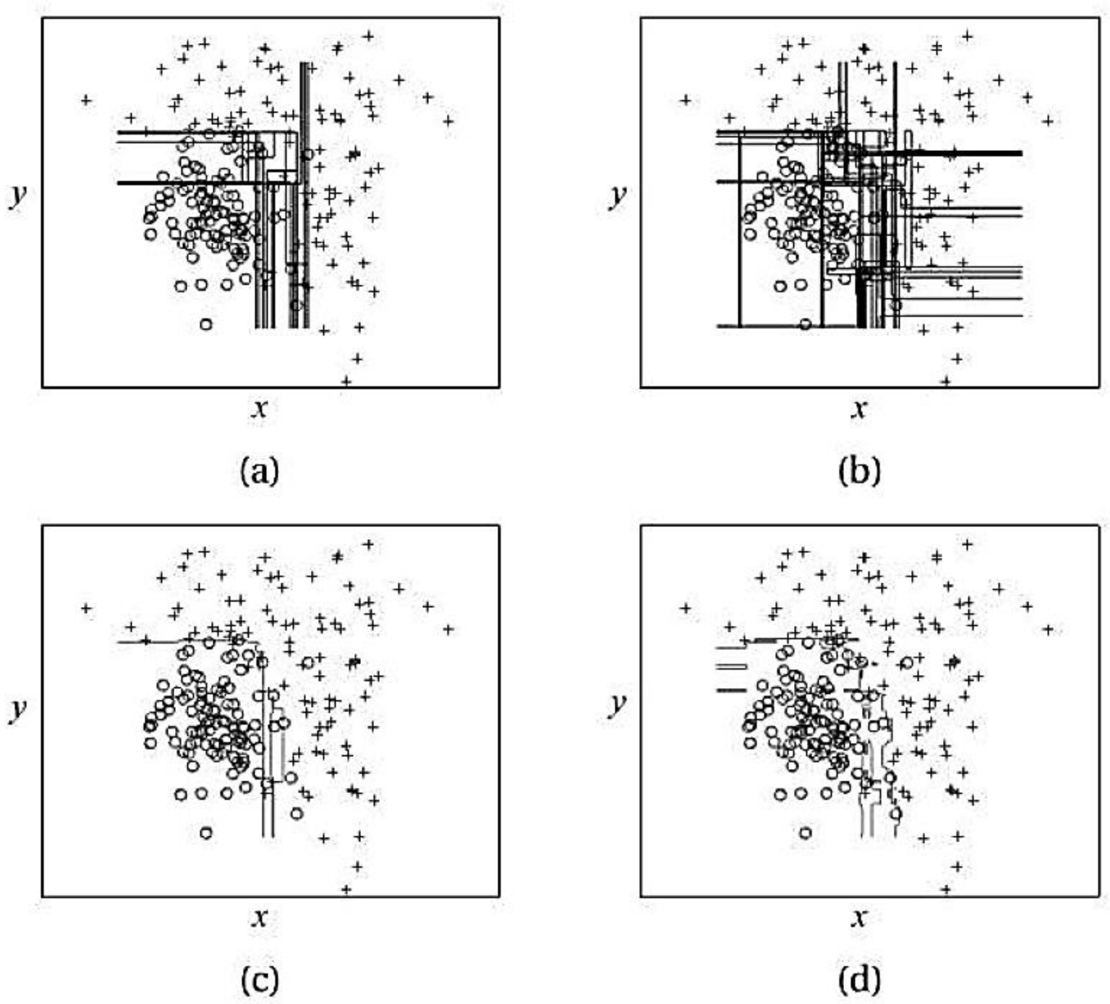
- Usual approach is to build very deep trees (low bias)
- Diversity in individual tree prediction functions comes from
  - bootstrap samples (somewhat different training data) and
  - randomized tree building

Effect of m size



From An Introduction to Statistical Learning, with applications in R (Springer, 2013)

Random forest examples

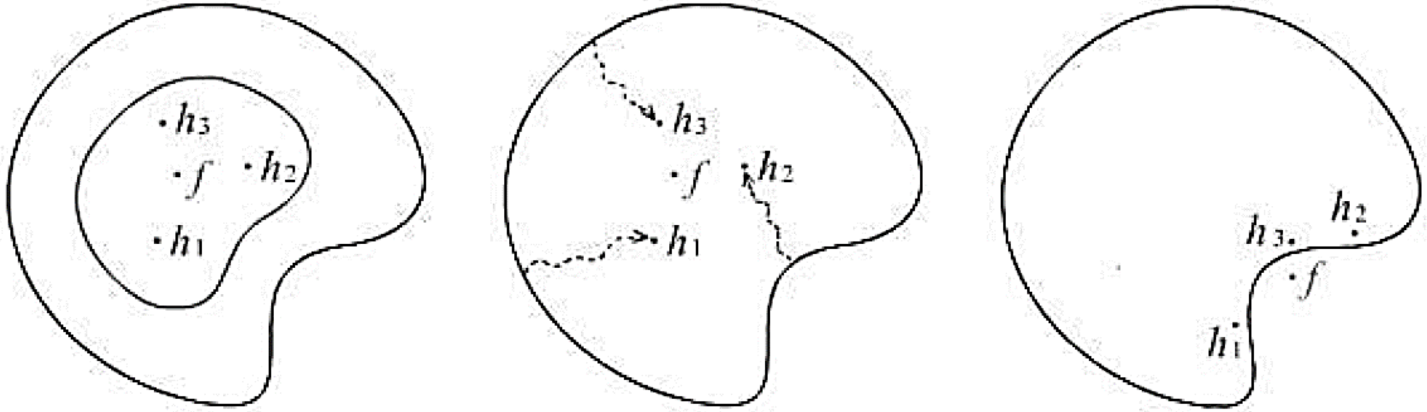


Decision boundaries on the three-Gaussians data set: (a) the 10 base classifiers of Bagging; (b) the 10 base classifiers of RF; (c) Bagging; (d) RF.

From Zhihua Zhou, Ensemble Methods: Foundations and Algorithms, 2012

Benefit of combination

- Statistical issue: the risk of choosing a wrong hypothesis can be reduced.
- Computational issue: the risk of choosing a wrong local minimum can be reduced.
- Representational issue: it maybe possible to expand the space of representable functions, and thus the learning algorithm may be able to form a more accurate approximation to the true unknown hypothesis.

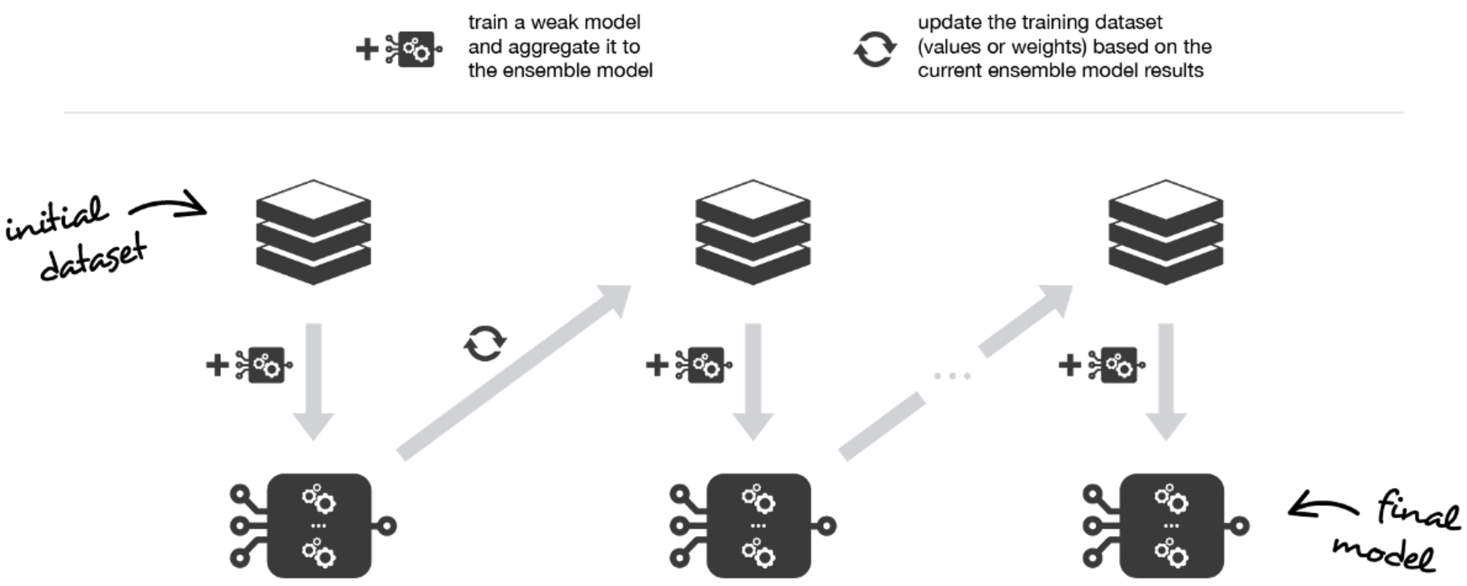


(a) Statistical (b) Computational (c) Representational

From Zhihua Zhou, Ensemble Methods: Foundations and Algorithms, 2012

Boosting

- Each model in the sequence is fitted giving more importance to observations in the dataset that were badly handled by the previous models in the sequence.
- As computations to fit the different models **can't be done in parallel** (unlike bagging), it could become too expensive to fit sequentially several complex models.



From Ensemble methods: bagging, boosting and stacking, Joseph Rocca

Diversity generation

- There is no generally accepted formal formulation and measures for ensemble diversity
- There are effective heuristic mechanisms for diversity generation in ensemble construction.
- The common basic idea is to inject some randomness into the learning process.
- Popular mechanisms include manipulating the data samples, input features, learning parameters, and output representations.

Diversity generation

- Data sample manipulation
- Given a data set, multiple different data samples can be generated, and then the individual learners are trained from different data samples.
- Generally, the data sample manipulation is based on sampling approaches, e.g., Bagging adopts bootstrap sampling, AdaBoost adopts sequential sampling, etc.
- Input features manipulation
- Different subsets of features provide different views on the data.
- Individual learners trained from different subsets of features are usually diverse. E.g. Random forests

Diversity generation

- Learning parameters manipulation
- Generate diverse individual learners by using different parameter settings for the base learning algorithm.
- For example, different initial weights or regularization terms for neural networks, different split selections for decision trees, etc.
- Output representations manipulation
- Generate diverse individual learners by using different output representations.
- For example, randomly changes the labels of some training instances, converts multi-class outputs to multivariate regression outputs to construct individual learners, etc.

References

- Zhihua Zhou, Machine learning, 2016
- Zhihua Zhou, Ensemble Methods: Foundations and Algorithms, 2012
- Gareth James, et al. An introduction to statistical learning, 2013

https://faculty.marshall.usc.edu/gareth-james/ISL/ISLR%20Seventh%20Printing.pdf

- A. Criminisi, et al. Decision Forests for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning, 2016

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/decisionForests_MSR_TR_2011_114.pdf

Q&A