Decision Tree Ensembles

Bagging, Random Forest, Boosting

Dr. Ratna Babu Chinnam

Industrial & Systems Engineering

Wayne State University

Primary Sources: O. Tastan of CMU | P. Protopapas, K. Rader, W. Pan of Harvard | Hastie et al. Link

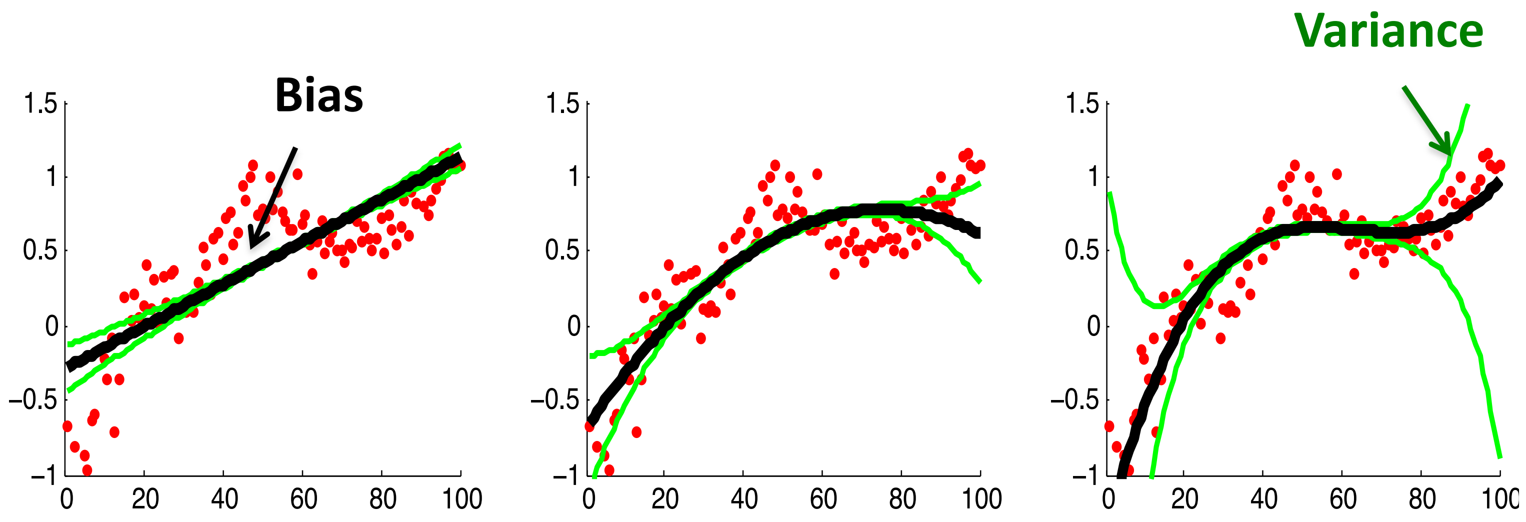
Need for Ensemble Methods

- A single decision tree does not perform well
- But, it is super fast
- What if we learn multiple trees?
- Need to make sure they do not all just learn the same
- Need to also combine the results to produce the final output

Bias/Variance Tradeoff

- Ensemble methods that **minimize variance**
  - Bagging
  - Random Forests

- Ensemble method that **minimize bias**
  - Boosting



Bagging

Bootstrap aggregating

Bagging: *Bootstrap aggregating*

- Technique for splitting data for **reducing model variance**
  - Classification: *committee* of models each cast a vote for predicted class
  - Regression: take average of estimates from individual learners
- Overall, improves accuracy by reducing overfitting (variance)
- Normally uses one type of classifier (decision trees are popular)
- Easy to parallelize

Bagging

- **Ideal Setting**: Many training sets S'
  - Train model using each S'
  - Average predictions

Population $P(x, y)$

Person	Age	Male?	Height > 55”
James	11	1	1
Jessica	14	0	1
Alice	14	0	1
Amy	12	0	1
Bob	10	1	1
Xavier	9	1	0
Cathy	9	0	1
Carol	13	0	1
Eugene	13	1	0
Rafael	12	1	1
Dave	8	1	0
Peter	9	1	0
Henry	13	1	0
Erin	11	0	0
Rose	7	0	0
Iain	8	1	1
Paulo	12	1	0
Margaret	10	0	1
Frank	9	1	1
Jill	13	0	0
Leon	10	1	0
Sarah	12	0	0
Gena	8	0	0
Patrick	5	1	1

⋮

Sample S'

Person	Age	Male?	Height > 55”
Alice	14	0	1
Bob	10	1	1
Carol	13	0	1
Dave	8	1	0
Erin	11	0	0
Frank	9	1	1
Gena	8	0	0

- **Variance reduces linearly**
- **Bias unchanged**

“Sampled” independently

- **In Practice:** Resample S' with replacement (bootstrap aggregating)
  - Not all data points enter resampled datasets (same size as original set)
  - Some data points are selected multiple times
  - Train model using each S'
  - Average predictions

Sample S

Person	Age	Male?	Height > 55”
Alice	14	0	1
Bob	10	1	1
Carol	13	0	1
Dave	8	1	0
Erin	11	0	0
Frank	9	1	1
Gena	8	0	0

Sample S'

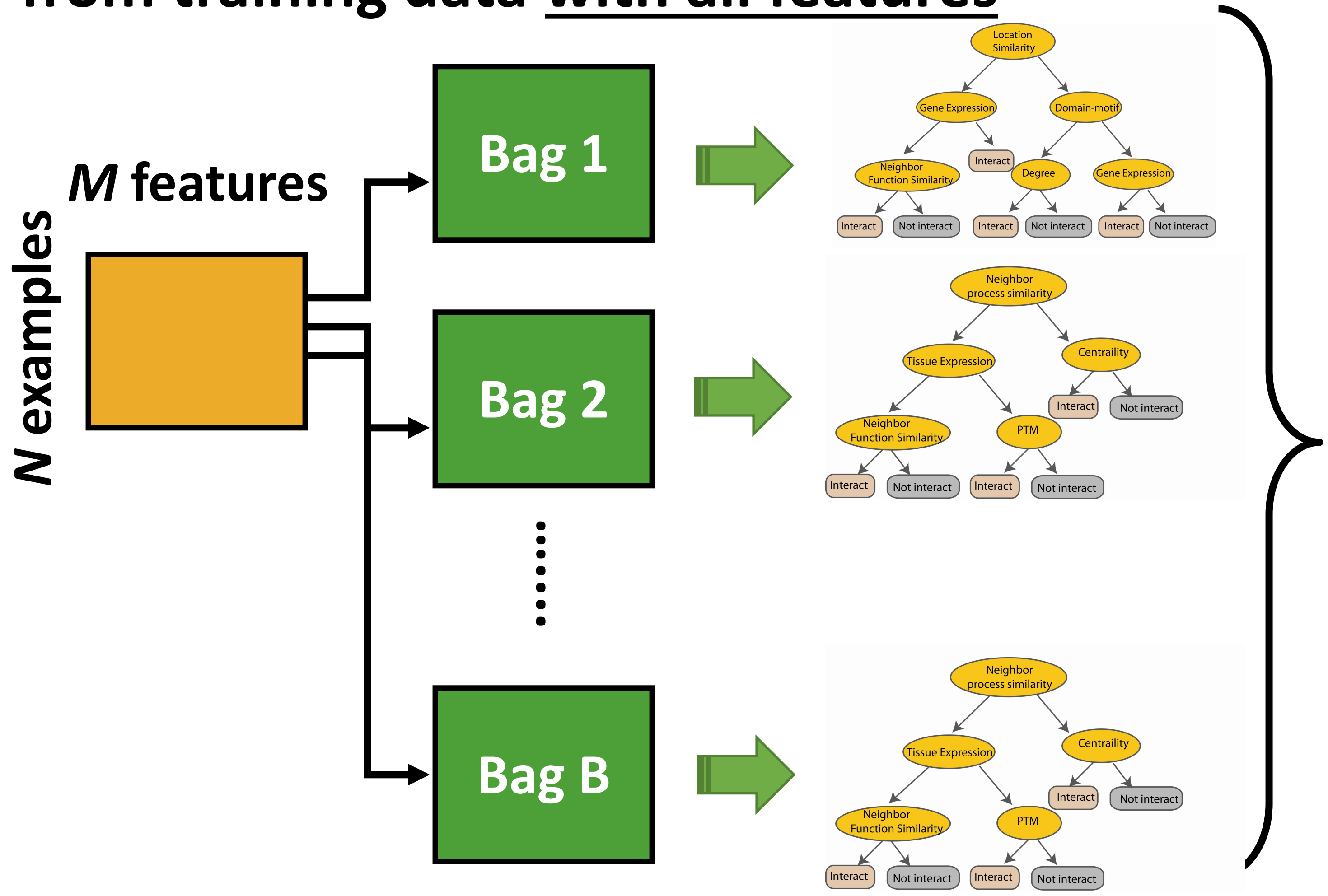
Person	Age	Male?	Height > 55”
Alice	14	0	1
Bob	10	1	1
Carol	13	0	1
Dave	8	1	0
Erin	11	0	0
Frank	9	1	1
Gena	8	0	0

- **Variance reduces sub-linearly (S' are correlated)**
- **Bias often increases slightly**

“Resample” with replacement

Bagging: Classification

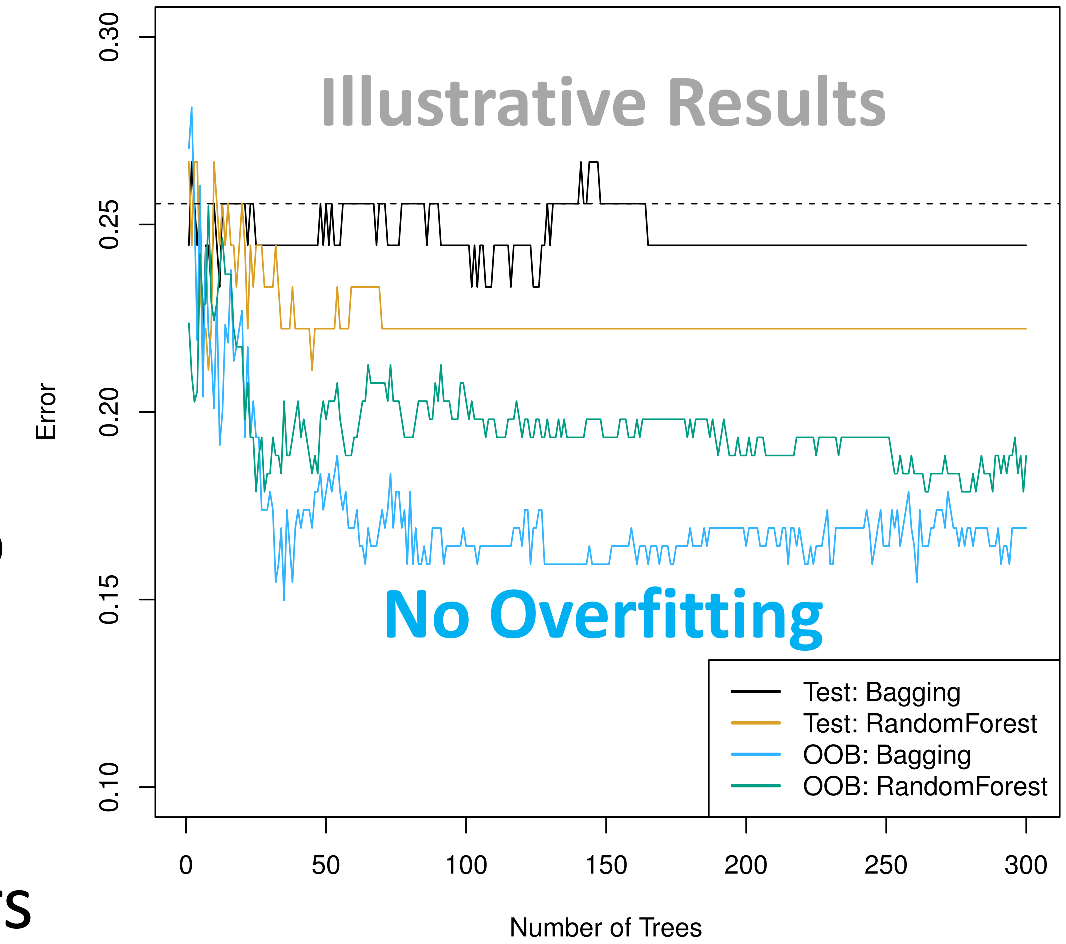
Create “bootstrap” samples
from training data <u>with all features</u>



Take the majority vote!

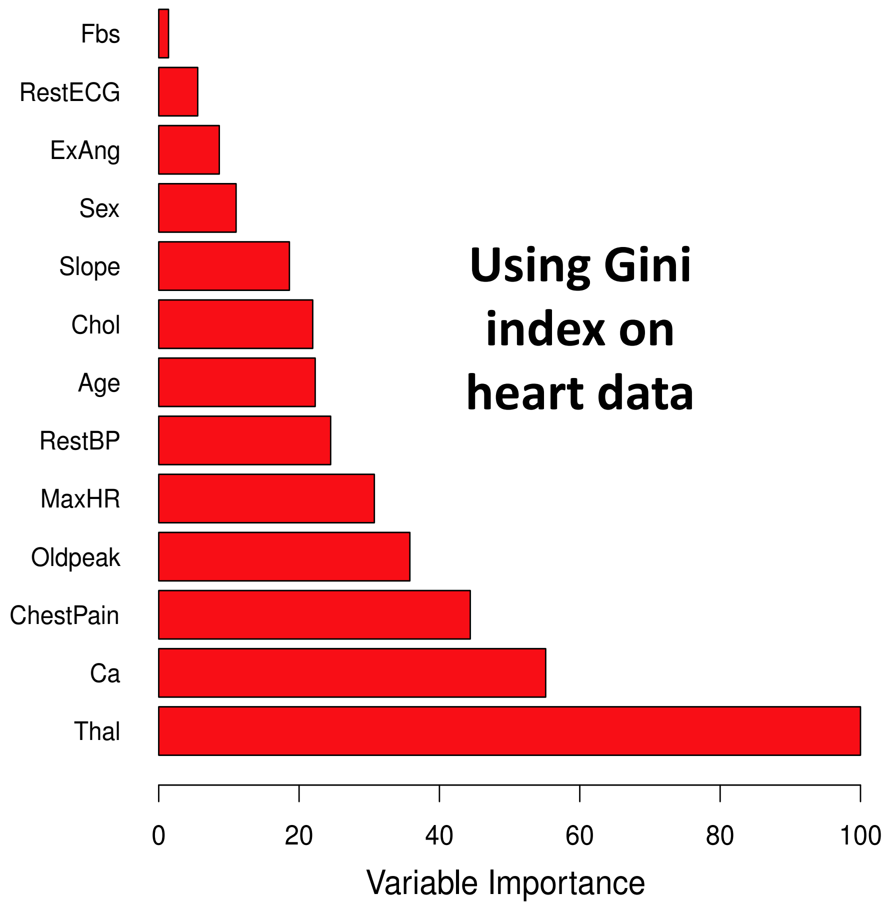
Out-of-Bag (OOB) Error Estimation

- No cross validation!
- Not all observations are used for each bootstrap sample
- Left out observations are called out-of-bag samples (OOB)
- We can predict response for i-th observation using each of the trees in which that observation was OOB and do this for all n observations
- Calculate overall OOB MSE or classification error
- Build bags sequentially: Once OOB errors stabilize, stop building!



Bagging: Variable Importance Measures

- **Difficult to interpret resulting model**
  - Improves prediction accuracy at the expense of interpretability
- Calculate performance improvement due to splits over a given predictor, averaged over all trees
  - RSS (Residual sum of squares) for bagging regression trees
  - Gini index for bagging classification trees



Using Gini index on heart data

Bagging: Issues

- Each tree is identically distributed (i.d.) not i.i.d.
  - All features are employed with similar datasets
- Expectation of average of B such trees is same as expectation of any one of them
- **Bias of bagged trees is same as that of individual trees!**
- **Solution**: Randomly select a subset of features for every bag!
  - **Random Forest!**
  - **Decision trees from each bag are more independent!**

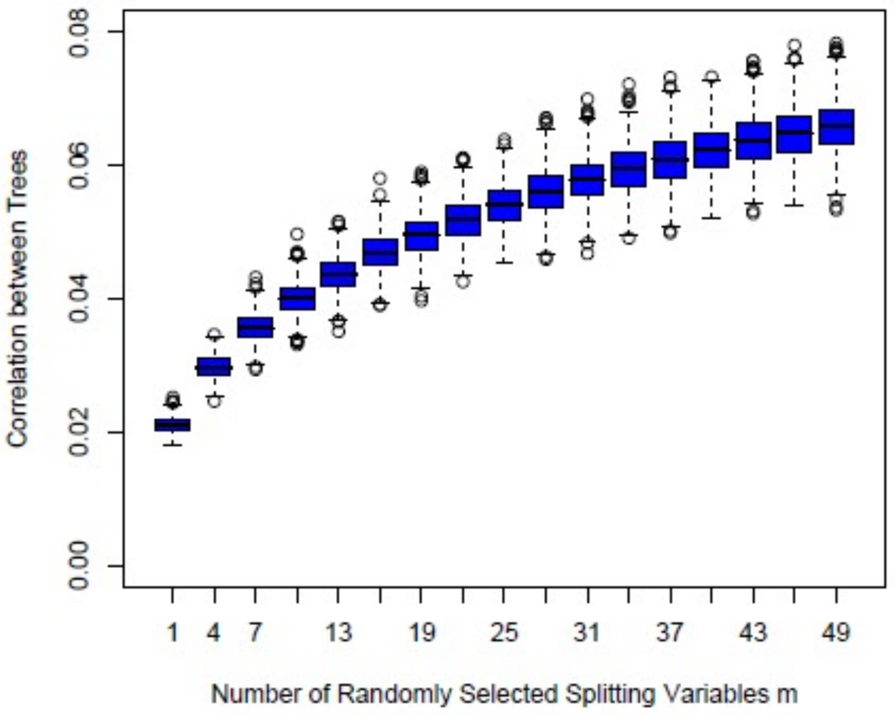
Random Forest

Also employs decision trees

Not all features are employed by each decision tree

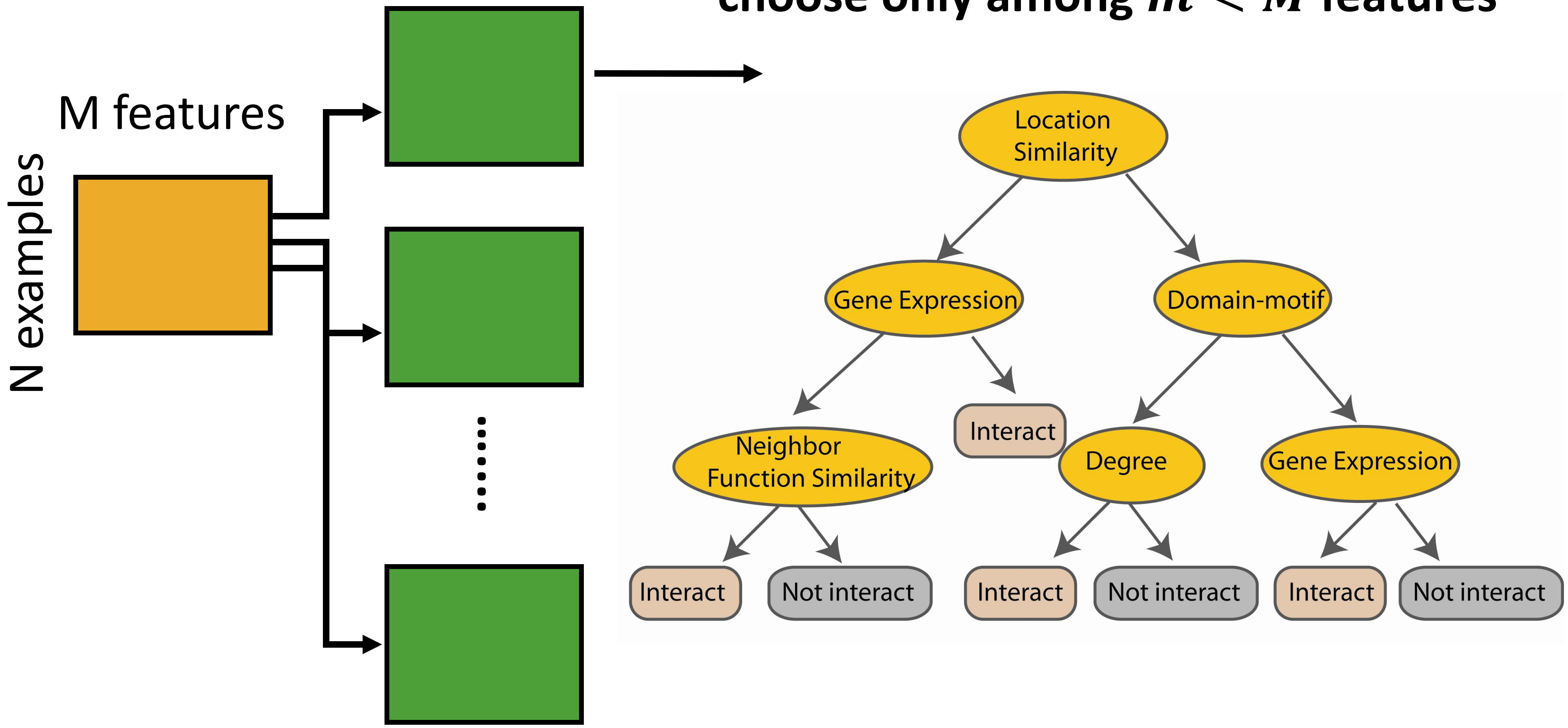
Random Forest

- **Random forest is an extension to bagging, which uses *de-correlated* trees**
- As in bagging, we build several decision trees on bootstrapped training samples
- **Each time a split in a tree is considered, a random sample of m predictors is chosen as candidates** from full set of M predictors
- Note that if $m = M$, then this is bagging!



Random Forest Classifier

At each node in choosing the split feature choose only among $m < M$ features



Random Forest Algorithm

For $b = 1\ to\ B$:

A. Draw a bootstrap sample Z^* of size N from training data by selecting m variables at random from M variables

B. Grow a random-forest tree to bootstrapped data, by recursively repeating following steps for each terminal node of tree, until minimum node size n_{min} is reached

   i. Pick best variable/split-point among the m variables

   ii. Split node into daughter nodes

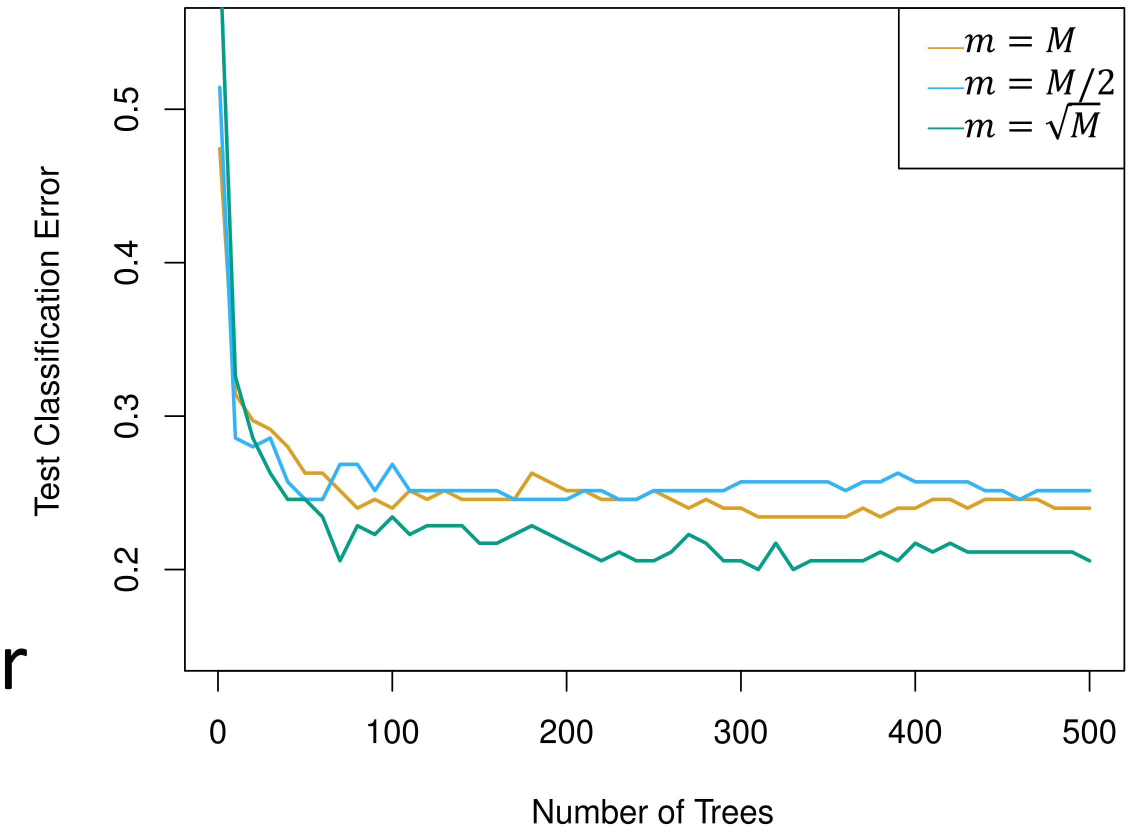
C. Output the ensemble of trees!

Random Forest Tuning

- The inventors make the following recommendations:
  - For classification, default value for m is $\sqrt{M}$ and minimum node size is 1
  - For regression, default value for m is $\sqrt{M}/3$ and minimum node size is 5
- In practice, need to be treated as tuning parameters
- Like with Bagging, we can use OOB and therefore RFs can be fit in sequence, with cross-validation being performed along the way
- Once OOB error stabilizes, training can be terminated

Random Forest: Example

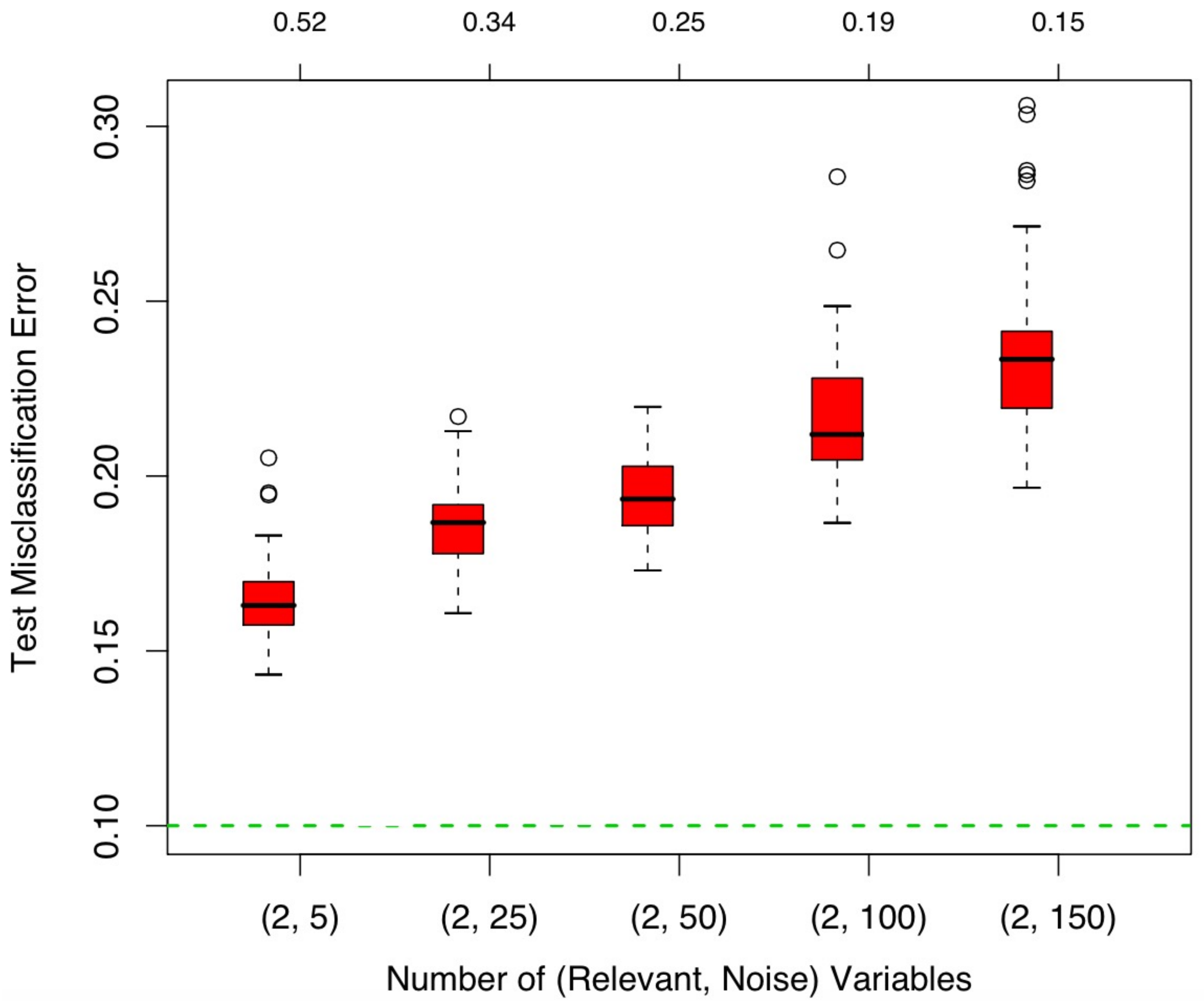
- 4,718 genes measured on tissue samples from 349 patients
  - Each gene has different expression
- Each of the patient samples has a qualitative label with 15 different levels: either normal or 1 of 14 different types of cancer
- Use random forest to predict cancer type based on 500 genes that have the largest variance in training set



Random Forest Issues

- When number of variables is large, but fraction of relevant variables is small, random forests are likely to perform poorly when m is small. Why?
- Because: At each split the chance can be small that the relevant variables will be selected
- For example, with 3 relevant and 100 not so relevant variables the probability of any of the relevant variables being selected at any split is ~ 0.25

Probability of being selected



RF: Variable Importance Measures

- Record prediction accuracy on OOB samples for each tree
- Randomly permute data for column j in OOB samples and record accuracy again
- Decrease in accuracy as a result of this permuting is averaged over all trees, and is used as a measure of variable importance

Boosting

Boosting is a general approach that can be applied to many statistical learning methods for regression or classification

Boosting

- **Bagging**: Generate multiple trees from bootstrapped data and average the trees. Results in i.d. trees and not i.i.d.
- **RF**: Produces i.i.d. (or more independent) trees by randomly selecting a subset of predictors at each step
- **Boosting**: Works very differently
  1. Boosting does not involve bootstrap sampling
  2. **Trees are grown sequentially**: Each tree is grown using information from previously grown trees
  3. Like bagging, boosting involves combining a large number of decision trees, $f^1, \ldots, f^B$

Boosting: Sequential Fitting

WAYNE STATE UNIVERSITY

- Given current model, **fit a decision tree to residuals from model**
  - Response variable now is residuals and not Y
- We then **add this new decision tree into the fitted function in order to update the residuals**
- Learning rate must be controlled
  - Shrinkage parameter λ

Boosting for Regression

1. Set $f(x) = 0$ and $r_i = y_i$ for all i in the training set

2. For $b = 1,2, \dots, B$, repeat:

   a. Fit a tree with d splits (+1 terminal nodes) to training data (X, r)

   b. Update tree by adding in a shrunken version of new tree:

$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$$

   c. Update residuals:

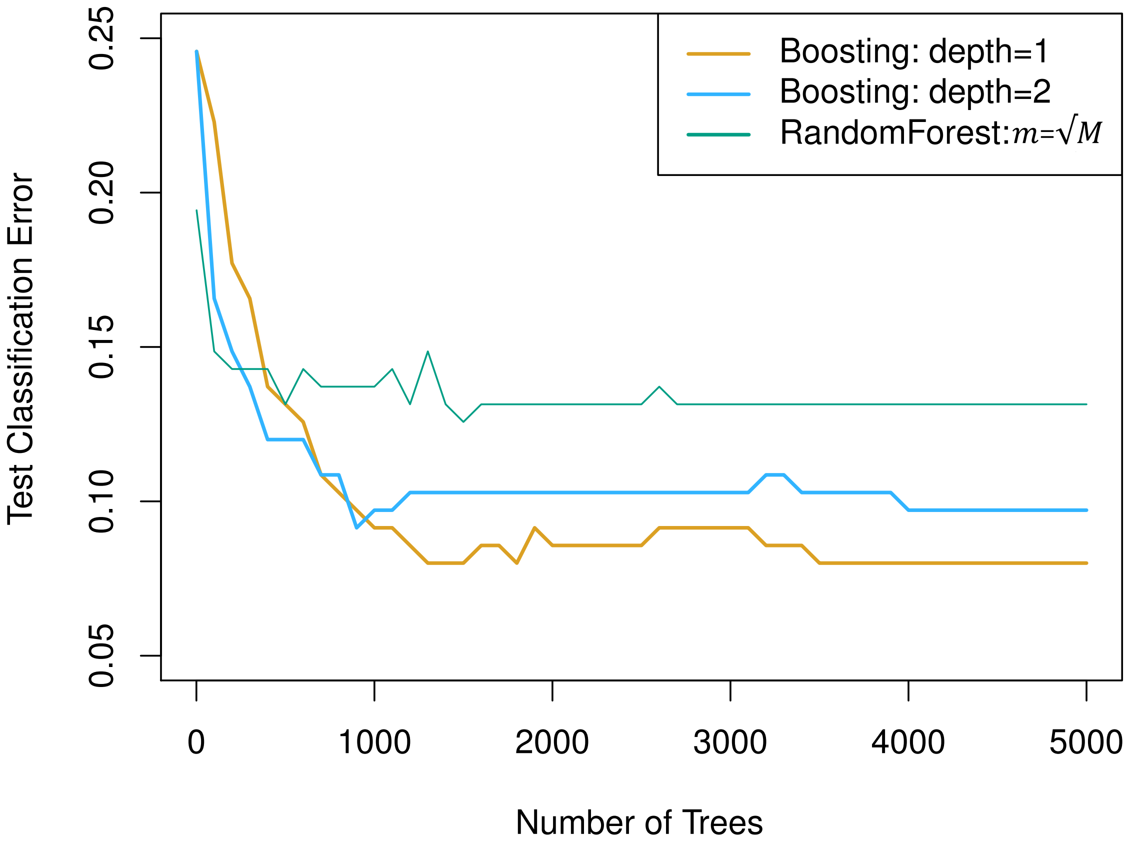
$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$$

3. Output boosted model:

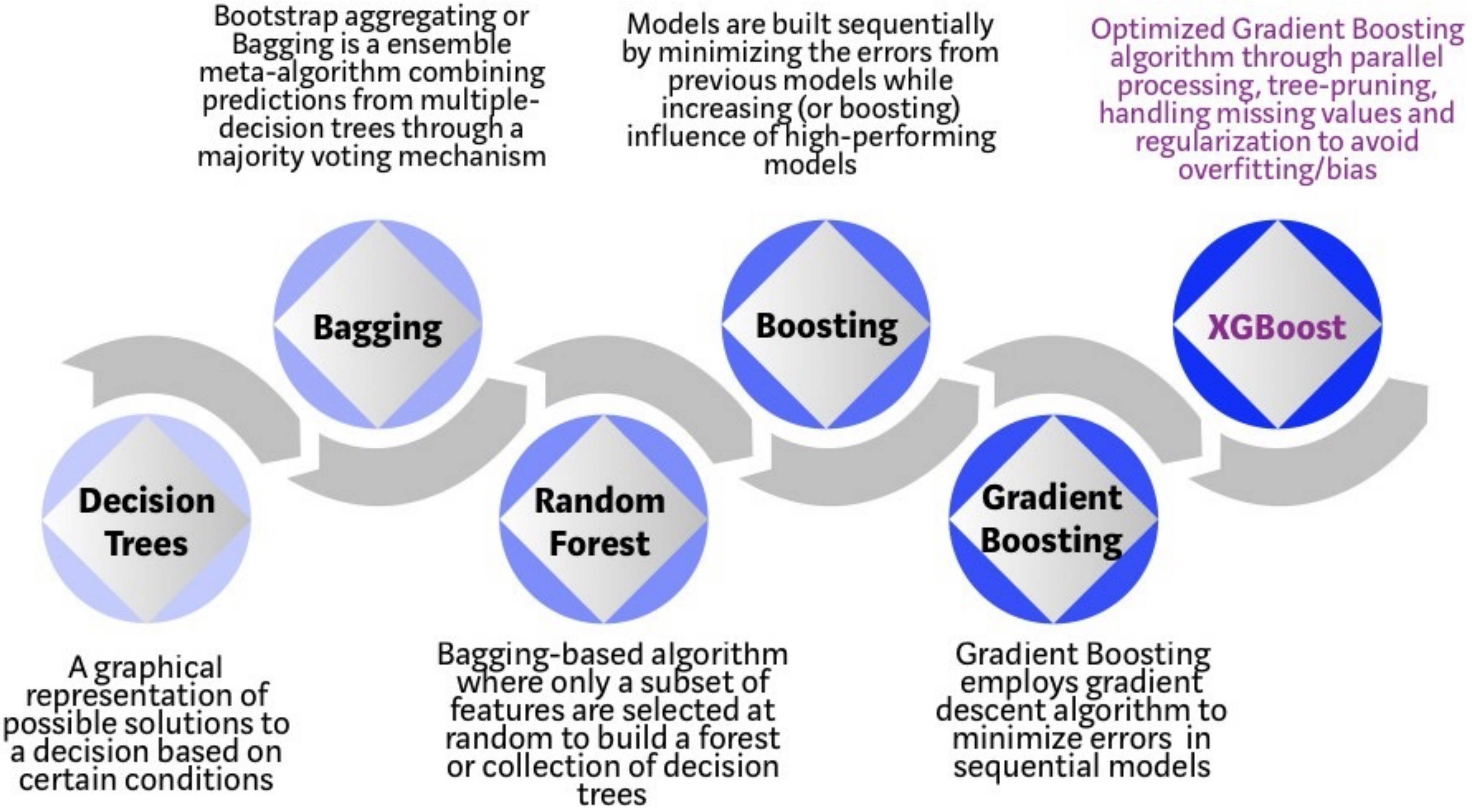
$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x)$$

Boosting Tuning Parameters

- Number of trees B:
  - RF and Bagging do not overfit as B increases
  - Boosting can overfit! Cross Validation
- Shrinkage Parameter λ:
  - Typical values are 0.01 or 0.001 but depends on problem
  - λ only controls learning rate
- Number of splits in each tree d:
  - Controls complexity of boosted ensemble
  - Stumpy trees, $d = 1$ can work well



Ensemble Tree Models



Source: Vishal Morde | Link