Ensemble methods – Bagging and Random Forest

Bin Li

IIT Lecture Series

Model selection vs. model combination

- Model selection:
  - Many models, which one to **choose**?
  - Goal: good interpretability and/or better predictive performance.
- Model combination:
  - Many models, how to **combine**?
    - Equally averaging (bagging, random forest).
    - Exponentially weighted model averaging (adaboost)
    - Weight optimization using stacking.
  - Goal: Better predictive performance.
- Takeaway message:
  - Model selection: works better if one model is significantly more accurate than other models – no ambiguity of which single model is better.
  - Equally weighted averaging: works better if all models have similar prediction accuracy, but are different – some ambiguity of which single model is better.

Rashomon effects

- Rashomon is a wonderful Japanese movie in which four people, from different vantage points, witness an incident in which one person dies and another is supposedly raped. When they come to testify in court, they all report the same facts, but their stories of what happened are very different.
- In statistics, "Rashomon Effect" means there is often a multitude of different models giving about the same minimum error rate.
- The effect is most obvious in selecting the best model on high dimensional data, such as subset selection in linear regression.

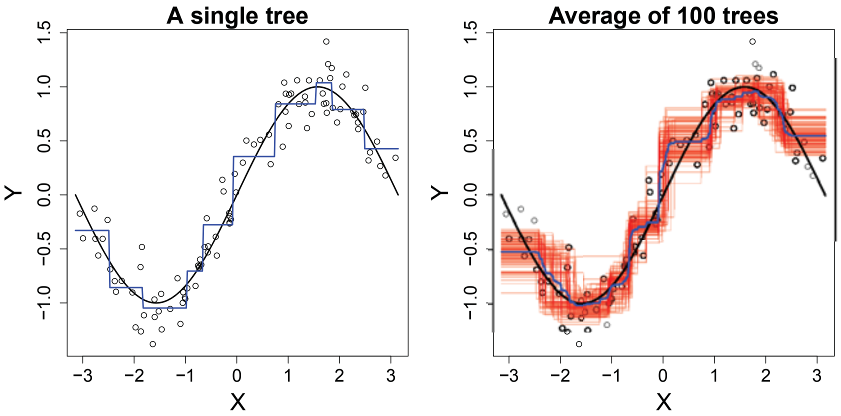
Figure from wikipedia.org.

Rashomon effects (cont.)

- Suppose there are 30 variables and we want to find the best five variable linear regressions.
- There are about 140,000 five-variable subsets in competition. Usually we pick the one with the lowest RSS (on training or test set if available). But generally there are many five-variable equations that have RSS within 1.0% of the lowest RSS. For example (see Breiman 1996):
  - Picture 1: $y = 2.1 + 3.8x_3 - 0.6x_8 + 83.2x_{12} - 2.1x_{17} + 3.2x_{27}$
  - Picture 2: $y = -8.9 + 4.6x_5 + 0.01x_6 + 12.0x_{15} + 17.5x_{21} + 0.2x_{22}$
  - Picture 3: $y = -76.6 + 9.3x_2 + 22.0x_7 - 13.2x_8 + 3.4x_{11} + 7.2x_{28}$

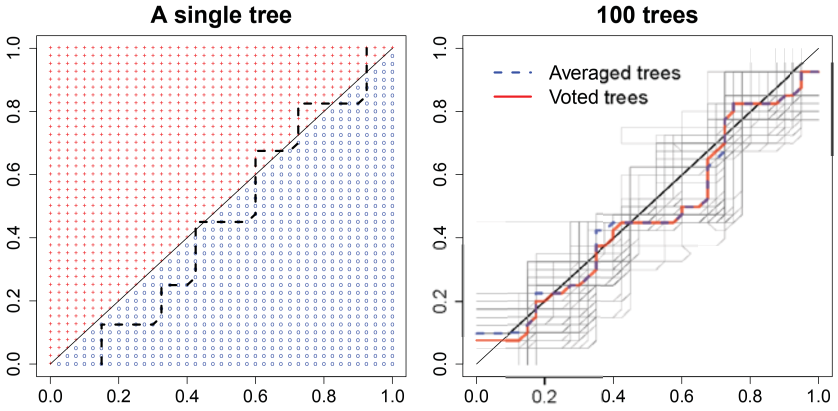
  Which one is better? The problem is that each one tells a different storyabout which variables are important.
- It also occurs with trees and neural nets, etc.
- The Rashomon Effect is closely related to model instability.

A simulation example of regression tree



- Single tree uses the whole dataset. RMSE: 0.152.
- 100 trees use bootstrap samples. RMSE: 0.130.

A simulation example of classification tree



- Single tree misclassification rate: 0.060.
- 100 trees misclassification rates: 0.034 (voted), 0.035 (averaged).

Bagging (**B**ootstrap **Agg**regat**ing**)

- Breiman, “Bagging Predictors”, *Machine Learning*, 1996.
- Fit classification or regression models to bootstrap samples from the data and combine by voting (classification) or averaging (regression/classification).
- When does it work?
  - Unstable models with similar performance.
    - Stable models: e.g. linear regression, logistic regression, pruned trees.
    - Unstable models: e.g. unpruned trees, neural network, subset selection in regression and classification.
  - Model is a nonlinear or adaptive function of the data.
  - Parametric bootstrapping in previous example converges to original fit with $B \to \infty$.
- How does it work? Variance reduction through averaging. No effects on bias.

Variance reduction in Bagging

- Set of estimators: $\hat{f}_1(x), \hat{f}_2(x), \ldots, \hat{f}_k(x)$
- Simple average: $\hat{f}^B(x) = \frac{1}{k}\sum_{i=1}^{k} \hat{f}_i(x)$
- Variance:

$$\begin{aligned} Var(\hat{f}^B(x)) &= \mathbf{E}\left(\hat{f}^B(x) - \mathbf{E}\hat{f}^B(x)\right)^2 \\ &= \mathbf{E}\left[\frac{1}{k^2}\left(\sum_{i=1}^{k}\hat{f}_i(x) - \mathbf{E}\hat{f}_i(x)\right)^2\right] \\ &= \frac{1}{k^2}\sum_{i=1}^{k}\mathrm{Var}\left\{\hat{f}_i(x)\right\} + \frac{1}{k^2}\sum_{i\neq j}\mathrm{Cov}\left\{\hat{f}_i(x), \hat{f}_j(x)\right\} \end{aligned}$$

- Assume:
  Covariance constant: $\mathrm{Cov}\left\{\hat{f}_i(x), \hat{f}_j(x)\right\} \approx \rho\sigma^2$

  Variance constant: $\mathrm{Var}\left\{\hat{f}_i(x)\right\} \approx \sigma^2$

  Then: $Var(\hat{f}^B(x)) \approx \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$

Example: bagging with trees

- Generate a sample of size $N = 30$, with two class and $p = 5$ features
- Each feature is generated from $N(0, 1)$
- Pairwise correlation is 0.95
- The response Y was generated according to

$$\Pr(Y = 1|x_1 \leq 0.5) = 0.2 \quad \text{and} \quad \Pr(Y = 1|x_1 > 0.5) = 0.8$$

- Bayes error is 0.2
- Test sample size is 2000
- Classification trees were fitted on training data and 200 bootstrap samples

Example: bagging with trees (cont.)

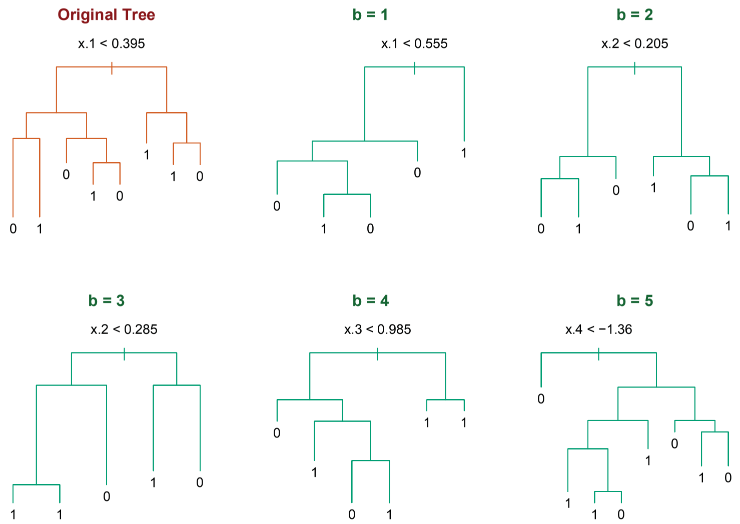


Figure from EOSL 2009.

Example: bagging with trees (cont.)

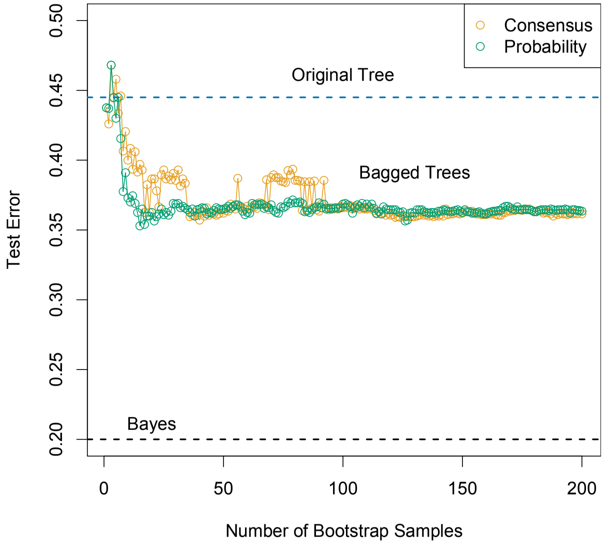


Figure from EOSL 2009.

Why bagging trees and bagging in general

- Trees can capture complex interaction structures in the data.
- Trees can handle mix-type of data (categorical and numerical) and miss values.
- With large depth, trees have relatively low bias but high variance.
- If the underlying $f(X)$ is smooth, bagging tends to reduce bias by "sanding off" the corners of the step functions.
- Bagging is NOT limited to tree methods and bootstrap samples (Buja and Stuetzle, 2006).
- Sometimes (not often) bagging does not work:
  - Bagging is unstable with respect to the training set when extreme outliers exist. It may actually increase variance.
  - Bag a stable model may make it worse.
  - sometime bagging can make bias worse (e.g. $f(x)$ is a step function).

Bagging CART on benchmark datasets (Breiman, 1996)

- Classification problems:

Dataset	# cases	# vars	# classes	CART(%)	Bagged CART(%)	Decrease (%)
Waveform*	300 (1800)	21	3	29.1	19.3	34
Heart	1395	16	2	4.9	2.8	43
Breast Cancer	699	9	2	5.9	3.7	37
Ionosphere	351	34	2	11.2	7.9	29
Diabetes	768	8	2	25.3	23.9	6
Glass	214	9	6	30.4	23.6	22
Soybean	683	35	19	8.6	6.8	21

- Regression problems:

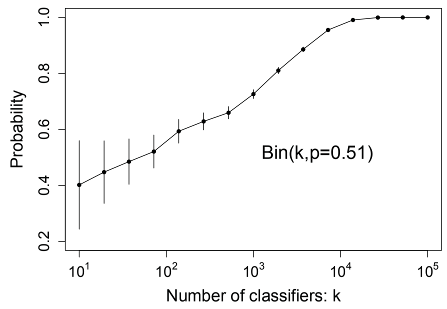
Dataset	# cases	# vars	CART	Bagged CART	Decrease
Boston housing	506	12	20.0	11.6	42%
Ozone	330	8	23.9	18.8	21%
Friedman #1*	200(1000)	10	11.4	6.1	46%
Friedman #2*	200(1000)	10	31,100	22,100	29%
Friedman #3*	200(1000)	10	0.0403	0.0242	40%

- For all real datasets, 90% randomly selected as training set, rest as test set. All the results are based on 100 random splits.
- For large data sets:

Dataset	Size	#Var	#classes	CART(%)	Bagged CART(%)	Decrease (%)
Letter	15,000(5,000)	16	26	12.6	6.4	49
Satellite	4435(2,000)	36	6	14.8	10.3	30
Shuttle	43,500(14,500)	9	7	0.062	0.014	77
DNA	2,000(1,186)	60	3	6.2	5.0	19

Wisdom of Crowds

- Assume weak classifiers are independent
- $S(x) \sim \text{Bin}(k, p)$ and $\Pr(S > k/2) \to 1$ as k gets large.
- Known as "Wisdom of Crowds" outside statistics (Surowiecki, 2004).



- Collective knowledge of a diverse and independent body of people typically exceeds the knowledge of any single individual, and can be harnessed by voting.

Random forest – a refinement of bagged trees

- Grow a forest of many trees. In `R` the `randomForest` package, the default number of trees is 500.
- Just like bagging, every tree in the random forest is grown on a bootstrap sample from the training data.
- At each node of a tree:
  - Randomly select m variables at random out of all M possible variables (independently for each node).
  - Find the best split from the randomly selected m variables (the splitting variable is often sub-optimal).
- Grow the trees to maximum depth without pruning.
- Use majority vote (classification) or average (regression) to get predictions for new data.
- Random forest tries to improve on bagging by further decorrelating the trees.
- Like bagging, improvement in prediction obtained by random forests is mainly a result of variance reduction.

Can a fully grown random tree be predictive?

- With a large number of predictors the eligible predictor set will be quite different from node to node.
- Important variables will make it into the tree (eventually).
- Explains in part why the trees must be grown out to absolute maximum full size
- A single tree in an RF forest can be predictive because it is a form of nearest neighbor classifier, one of the oldest and robust model-free machine learning technologies.
  - To reach a node in the tree a record must satisfy the condition at every parent (e.g. AGE>35, INCOME<60, EDUCATION>12, CITY=YES etc.)
  - Reaching a terminal node means being very similar to the training records that occupy that node.
  - The near neighbor predictor mechanism:
    - Find historical record that looks as similar as possible to the new record.
    - Predict that new record behaves just like historical record.

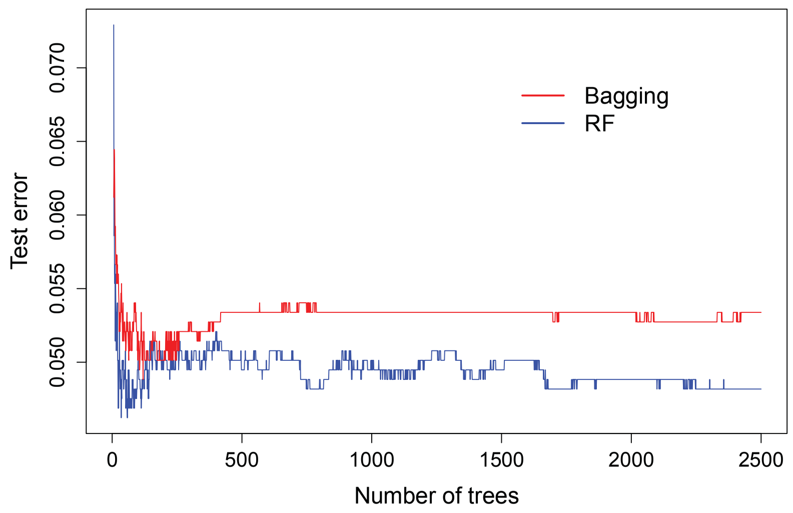
Spam email example

- Response: spam or normal email.
- Input variables: relative frequencies of 57 most commonly occurring words and punctuation marks in email message.
- Training set: 3065 obs (about 40% spams); test set: 1536 obs (about 39% spams).
- Average percentage of words or characters in an email message equal to the indicated word or character.

	george	you	hp	free	!	re	edu	remove
spam	0.00	2.26	0.02	0.52	0.51	0.13	0.01	0.28
email	1.27	1.27	0.90	0.07	0.11	0.42	0.29	0.01

- Objective: predict the email is spam or not.

Spam email example



Choice of m

The inventors make the following recommendation

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

Small m: low correlation (small variance) and large bias (if true f is complex).
Large m: high correlation and low bias.

An example: X_j and ϵ are all iid Gaussian. 500 training sets of size 100. One test set of size 600.

$$Y = \frac{1}{\sqrt{50}}\sum_{j=1}^{50} X_j + \epsilon$$

Choice of m (cont.)

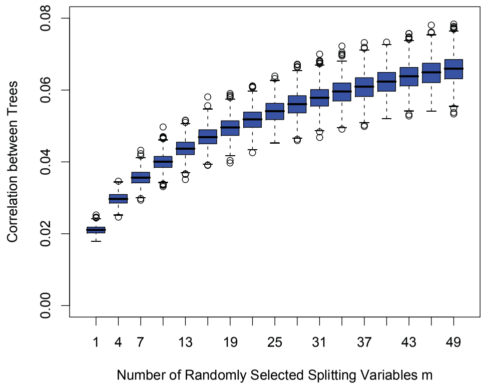


FIGURE 15.9. *Correlations between pairs of trees drawn by a random-forest regression algorithm, as a function of m. The boxplots represent the correlations at 600 randomly chosen prediction points x.*

Figure from EOSL 2009.

Choice of m (cont.)

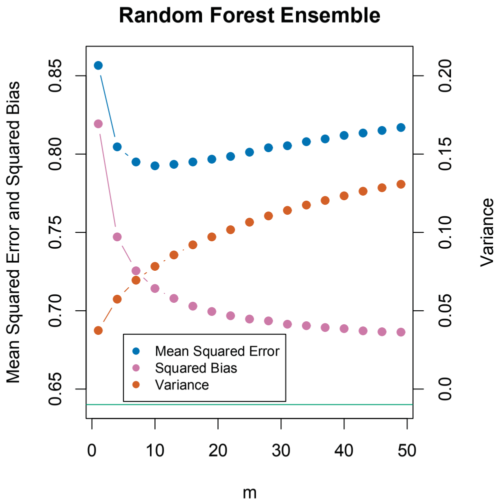


Figure from EOSL 2009.

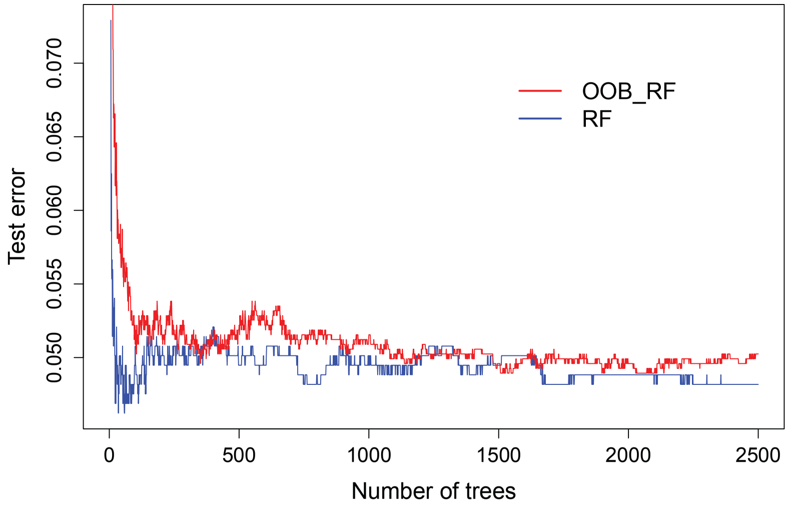
OOB error in random forest

- In bootstrapping, the probability of a case not being picked into the bootstrap sample is about one third.

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

- About 1/3 of the trees in RF did not use a particular case in fitting. We say that case is "out of bag" or "OOB" for those trees.
- In RF, the predictions of OOB samples from these trees provides an estimate of **generalization error** on new data, similar to K-fold CV.
- RF can be fit in one sequence, with "cross-validation" being performed along the way.
- Once the OOB error stabilizes, the training process can be terminated.
- OOB samples are also used to estimate relative variable importance.

OOB rrror in spam data



When RF performs poorly

- When the number of variables is large, but the fraction of relevant variables small, random forests are likely to perform poorly with small m. Because, at each split the chance can be small that the relevant variables will be selected.
- A simulation example: 50 simulations with a training sample of 300, and a test sample of 500

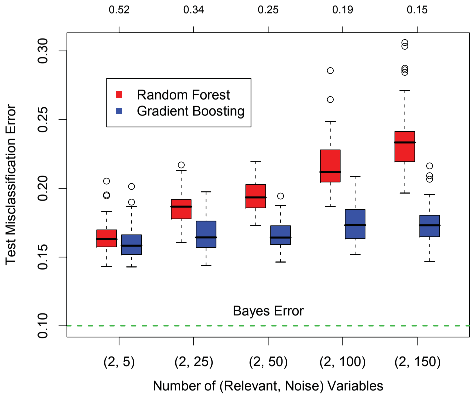


Figure from EOSL 2009.

Overfitting and the effect of tree size

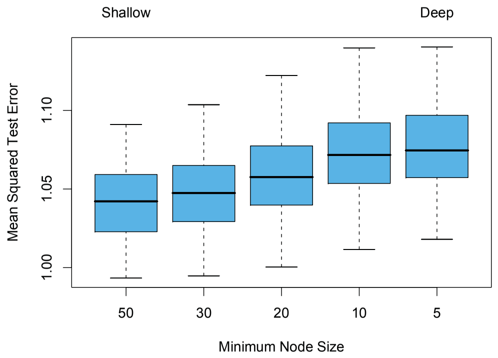


FIGURE 15.8. *The effect of* tree size *on the error in random forest regression. In this example, the true surface was additive in two of the 12 variables, plus additive unit-variance Gaussian noise. Tree depth is controlled here by the minimum node size; the smaller the minimum node size, the deeper the trees.*

Figure from EOSL 2009.

Measure of relative variable importance

- For a single decision tree $\mathcal{T}$, Breiman et al. (1984) proposed

$$\mathcal{I}_h^2(\mathcal{T}) = \sum_{t=1}^{T-1} \hat{i}_t^2 I(v(t) = h)$$

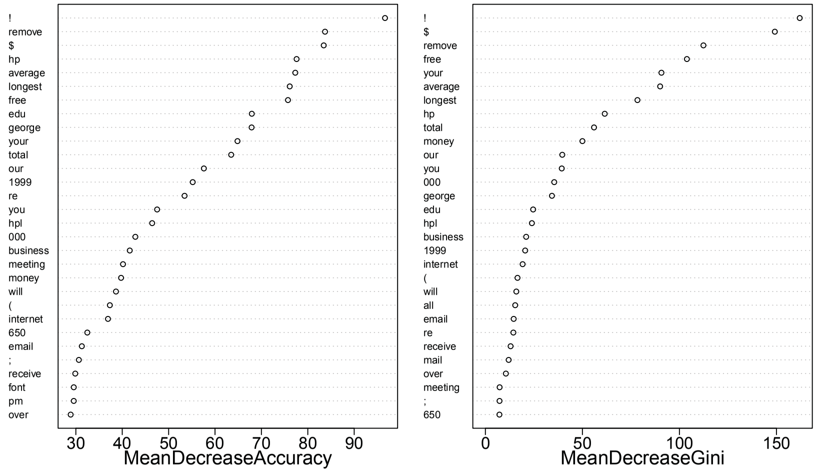
as a measure of relevance for each predictor variable X_h.

- The sum is over the $T - 1$ *internal nodes* of the tree.
- At each internal node, the variable chosen is the one that gives maximal estimated improvement $\hat{i}_t^2$ in squared error risk over that for a constant fit over the entire region.
- The squared relative importance of variable X_h is the sum of such squared improvements (weighted by node size) over all internal nodes for which it was chosen as the splitting variable.
- This importance measure is easily generalized to ensemble trees by simply averaged over the trees.

Another measure of variable importance

- Random forests also use the OOB samples to construct a different variable importance measure based on the prediction strength of each variable.
- When the bth tree is grown, the OOB samples are passed down the tree, and the prediction accuracy is recorded.
- The values for the jth variable are randomly permuted in the oob samples, and the accuracy is again computed.
- The decrease in accuracy as a result of this permuting is averaged over all trees, and is used as a measure of the importance of variable j in the random forest.
- The variable importance from the permutation approach is often more uniform over the variables.

Relative variable importance in Spam example



`MeanDecreaseAccuracy` is based on the permutation approach.

`MeanDecreaseGini` is based on the traditional rough-and-ready approach.

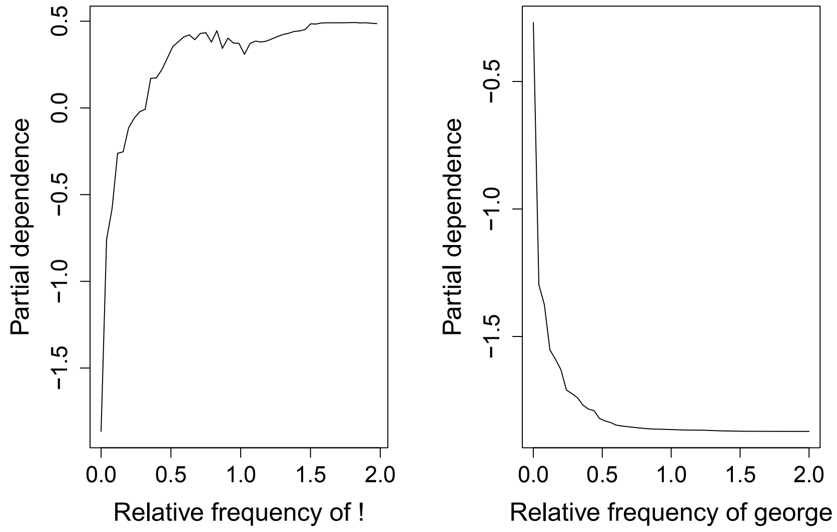
Partial dependence plot

- Consider $\mathcal{S} \subset \{1, 2, \ldots, p\}$ and $\mathcal{C}$ is the complement of $\mathcal{S}$. **Partial dependence** of $f(X)$ on $X_\mathcal{S}$ is its marginal average of f estimated by

$$\bar{f}_\mathcal{S}(X_\mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N} f(X_\mathcal{S}, x_{i\mathcal{C}})$$

  where $\{x_{i\mathcal{C}}\}_1^N$are the values of $X_\mathcal{C}$ occurring in data.
- Partial dependence function defined above represents the effect of $X_\mathcal{S}$ on $f(X)$ after accounting for the average effects of the other variables $X_\mathcal{C}$ on $f(X)$ (not ignoring $X_\mathcal{C}$).
- Useful when the variables in $X_\mathcal{S}$ do not have strong interactions with those in $X_\mathcal{C}$.
- The trend/shape of the partial dependence plot is more meaningful than the values on vertical axis.
- For K-classification, there are K partial dependence, one for each class. The plot can help reveal how the *log-odds* of realizing that class depend on the respective input variables.

Partial dependence plots in Spam example



Others

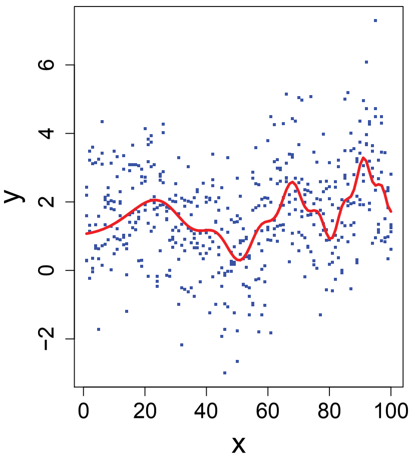
- The *proximity plot* gives an indication of which observations are effectively close together in the eyes of the RF.
- RF impute the missing values based on proximity measure.
- RF can also do the *unsupervised learning* such as clustering.
- `randomForest` package in `R`, maintained by Andy Liaw, available from the CRAN website
- `FORTRAN` code written by Leo Breiman and Adele Cutler is freely available at `http://www.math.usu.edu/~adele/forests/`
- The Weka machine learning archive `http://www.cs.waikato.ac.nz/ml/weka/` at Waikato University, New Zealand, offers a free `java` implementation of random forests

Other ideas to generate ensemble candidates

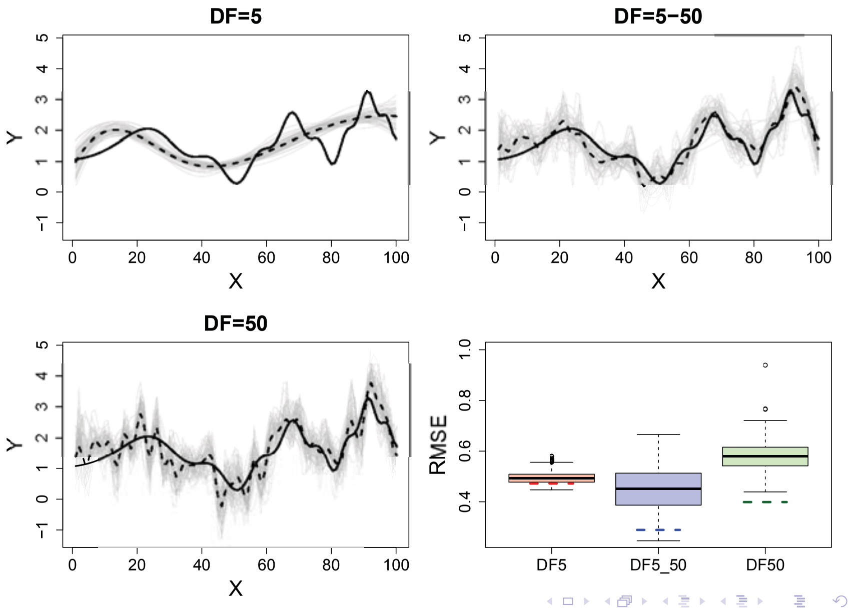
- Use different learning algorithms (e.g. different loss functions).
- Use different variations of data-processing.
  - different transformations of features
  - different subsets (selection) of features
- Use different tuning parameter configurations.
  - e.g. varying span values for loess smoother or K for nearest neighbor.
- Use randomization:
  - randomly select training data (e.g. bagging)
  - randomly select features or randomly select basis functions
  - randomization the algorithm especially if it contains some combinatoric elements (e.g. random forest).

A simulation example

- $y = f(x) + \epsilon$, where SNR=1/4. Function f is the red curve on the right plot.
- Data: randomly generate 500 obs (X has 100 levels from 1 to 100). Each level has 5 replicates.
- Fit a cubic splines on each bootstrap data with
  - Method 1: df=5 (i.e. 2 interior knots).
  - Method 2: randomly select df from a Unif[5, 50]
  - Method 3: df=50 (i.e. 47 interior knots)
- Bagging $B = 100$ times. Average the predicted values on $x = 1, \dots, 100$ for each method.
- RMSE for Method 1-3 are 0.47, 0.29 and 0.40, respectively.



A simulation example (cont.)



The success of ensemble methods

- "What are the best of the best techniques at winning Kaggle competitions?
  - Ensembles of decision trees
  - Deep learning

  account for 90% of top 3 winners – by Jeremy Howard, Chief Scientist of Kaggle, in KDD 2013.
- The first place winner of the "Best Classification Challenge" of the 2013 IEEE GRSS Data Fusion Contest used ensemble classification methods to combine multiple classifiers. See `http://hyperspectral.ee.uh.edu/?page_id=695`
- "Lessons from the Netflix Prize Challenge" by Robert Bell and Yehuda Koren.
  - "We found no perfect model. Instead, our best results came from combining predictions of models that complemented each other. While our winning entry, a linear combination of many prediction sets, achieved an improvement over Cinematch of 8.43%, the best single set of predictions reached only 6.57%."

Reference

- Bühlmann P. and B. Yu (2002) Analyzing Bagging, *The Annals of Statistics*, **30**, 927-961.
- Buja, A. and W. Stuetzle (2006) Observations on Bagging, *Statistica Sinica*, **16(2)**, 323-352.
- Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statistical Learning*, Springer.
- Leo Breiman (1996) Bagging predictors, *Machine Learning*, 24: 123-140.
- Leo Breiman (2001) Random forest, *Machine Learning*, 45: 5-32.