Model Building:
Ensemble Methods

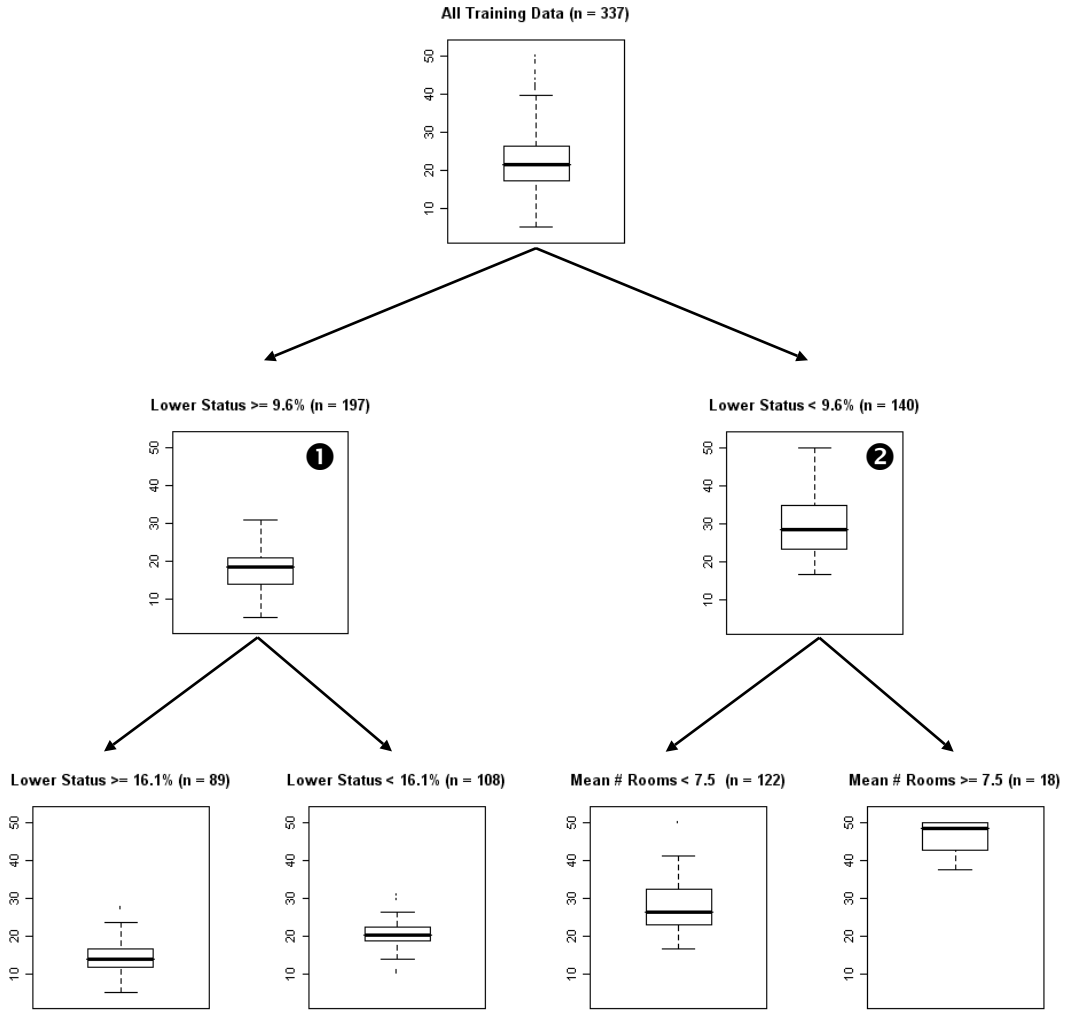
Max Kuhn and Kjell Johnson

Nonclinical Statistics, Pfizer

1

Splitting Example – Boston Housing

- Searching though the first left split (❶), the best split again uses the lower status %
- In the initial right split (❷), the split was based on the mean number of rooms
- Now, there are 4 possible predicted values



2

Single Trees

- Advantages
  - can be computed very quickly and have simple interpretations.
  - have built-in predictor selection: if a predictor was not used in any split, the model is completely independent of that data.
- Disadvantages
  - instability due to high variance: small changes in the data can drastically affect the structure of a tree
  - data fragmentation
  - high order interactions

3

Ensemble Methods

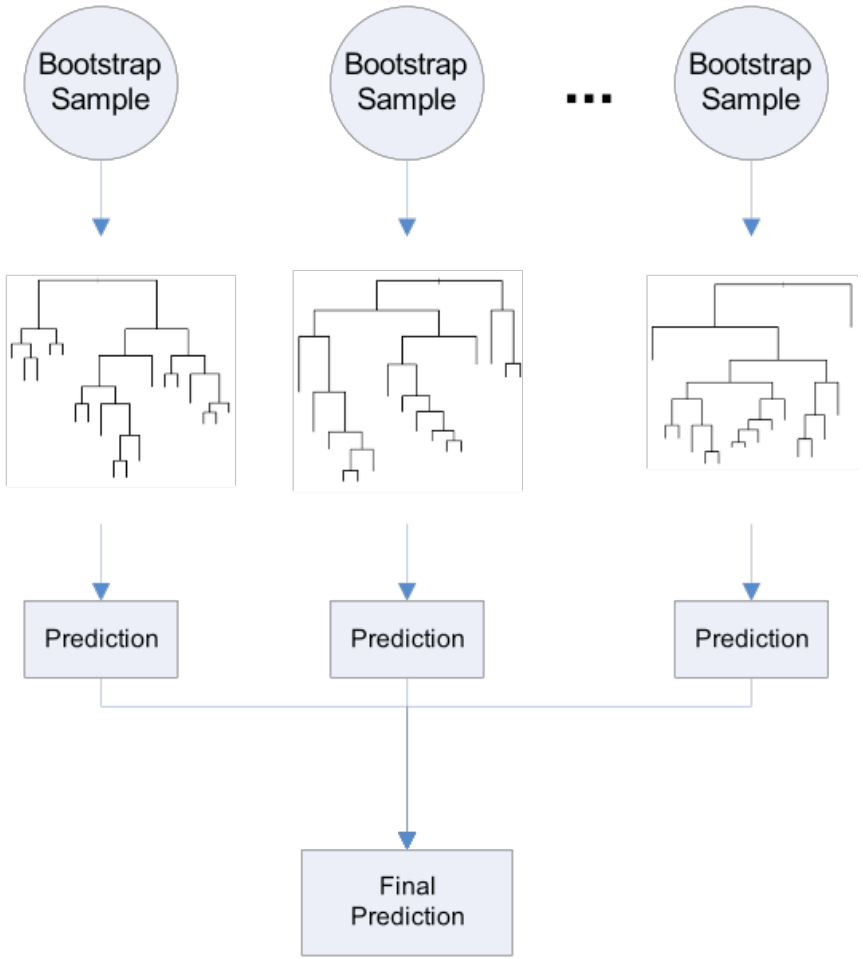
- Ensembles of trees have been shown to provide more predictive models than individual trees and are less variable than individual trees

- Common ensemble methods are:
  - Bagging
  - Random forests, and
  - Boosting

4

Bagging Trees

- Bootstrap Aggregation
  - Breiman (1994, 1996)
  - Bagging is the process of
    1. creating bootstrap samples of the data,
    2. fitting models to each sample
    3. aggregating the model predictions
  - The largest possible tree is built for each bootstrap sample



5

Bagging Model

Prediction of an observation, x:

$$F(x) = \frac{\sum_{m=1}^{M} f_m(\mathrm{x})}{M}$$

6

Comparison

- Bagging can significantly increase performance of trees
  - from resampling:

	Training Data (bootstrap)		Test	
	RMSE	**Q^2**	**RMSE**	**R^2**
Single Tree	5.18	0.700	4.28	0.780
Bagging	4.32	0.786	3.69	0.825

- The cost is computing time and the loss of interpretation
- One reason that bagging works is that single trees are unstable
  - small changes in the data may drastically change the tree

7

Random Forests

- Random forests models are similar to bagging
  - separate models are built for each bootstrap sample
  - the largest tree possible is fit for each bootstrap sample
- However, when random forests starts to make a new split, it only considers a random subset of predictors
  - The subset size is the (optional) tuning parameter
- Random forests defaults to a subset size that is the square root of the number of predictors and is typically robust to this parameter

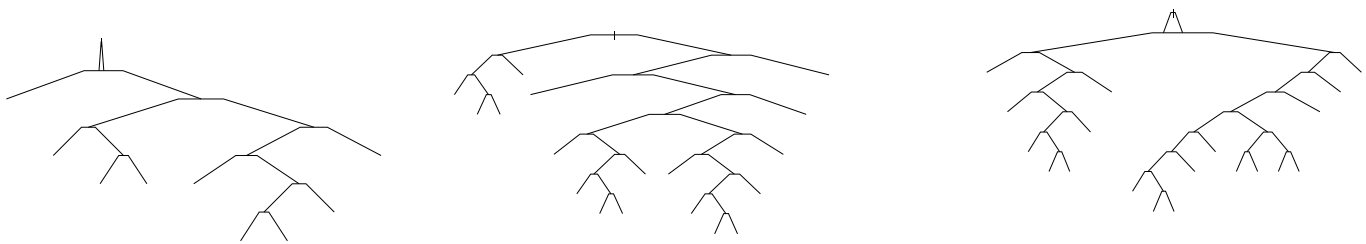
8

Random Predictor Illustration

Randomly select a subset of variables from original data

Dataset 1 | Dataset 2 | Dataset *M*

Build trees



Predict | Predict | Predict

Final Prediction

9

Random Forests Model

Prediction of an observation, x:

$$F(x) = \frac{\sum_{m=1}^{M} f_m(\mathrm{x})}{M}$$

10

Properties of Random Forests

- Variance reduction
  - Averaging predictions across many models provides more stable predictions and model accuracy (Breiman, 1996)
- Robustness to noise
  - All observations have an equal chance to influence each model in the ensemble
  - Hence, outliers have less of an effect on individual models for the overall predicted values

11

Comparison

- Comparing the three methods using resampling:

	Training Data (bootstrap)		Test	
	RMSE	**Q^2**	**RMSE**	**R^2**
Single Tree	5.18	0.700	4.28	0.780
Bagging	4.32	0.786	3.69	0.825
Rand Forest	3.55	0.857	3.00	0.885

- Both bagging and random forests are “memoryless”
  - each bootstrap sample doesn’t know anything about the other samples

12

Boosting Trees

- A method to “boost” weak learning algorithms (small trees) into strong learning algorithms
  - Kearns and Valiant (1989), Schapire (1990), Freund (1995), Freund and Schapire (1996a)

- Boosted trees try to improve the model fit over different trees by considering past fits

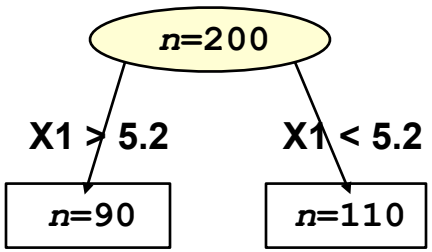
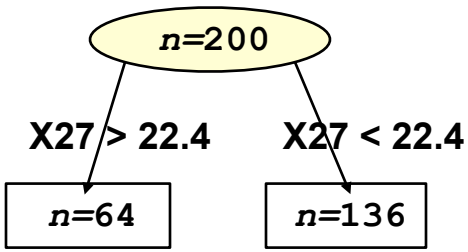
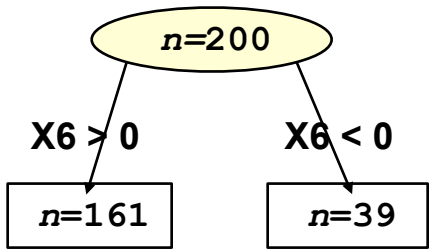
13

Boosting Trees

- First, an initial tree model is fit (the size of the tree is controlled by the modeler, but usually the trees are small (depth < 8))
  - if a sample was not predicted well, the model residual will be different from zero
  - samples that were predicted poorly in the last tree will be given more weight in the next tree (and vice-versa)
- After many iterations, the final prediction is a weighted average of the prediction form each tree

14

Boosting Illustration

Stage	**1**	**2**	**. . .**	***M***
Build weighted tree				
Compute error	$\sum_{i=1}^{n} e_i^2 = 32.9$	$\sum_{i=1}^{n} e_i^2 = 26.7$		$\sum_{i=1}^{n} e_i^2 = 29.5$
Compute stage weight	$\beta_{\text{stage 1}} = f(32.9)$	$\beta_{\text{stage 2}} = f(26.7)$		$\beta_{\text{stage M}} = f(29.5)$
Reweigh observations (w_i=1,2,..., n)	Determine weight of i^{th} observation: The larger the error, the higher the weight	Determine weight of i^{th} observation		

15

Boosting Trees

- Boosting has three tuning parameters:
  - number of iterations (i.e. trees)
  - complexity of the tree (i.e. number of splits)
  - learning rate: how quickly the algorithm adapts

- This implementation is the most computationally taxing of the tree methods shown here

16

Final Boosting Model

Prediction of an observation, x:

$$F(x) = \sum_{m=1}^{M} \left(\beta_m f_m \left(\mathrm{x} \right) \right)$$

where the β_m are constrained to sum to 1.

17

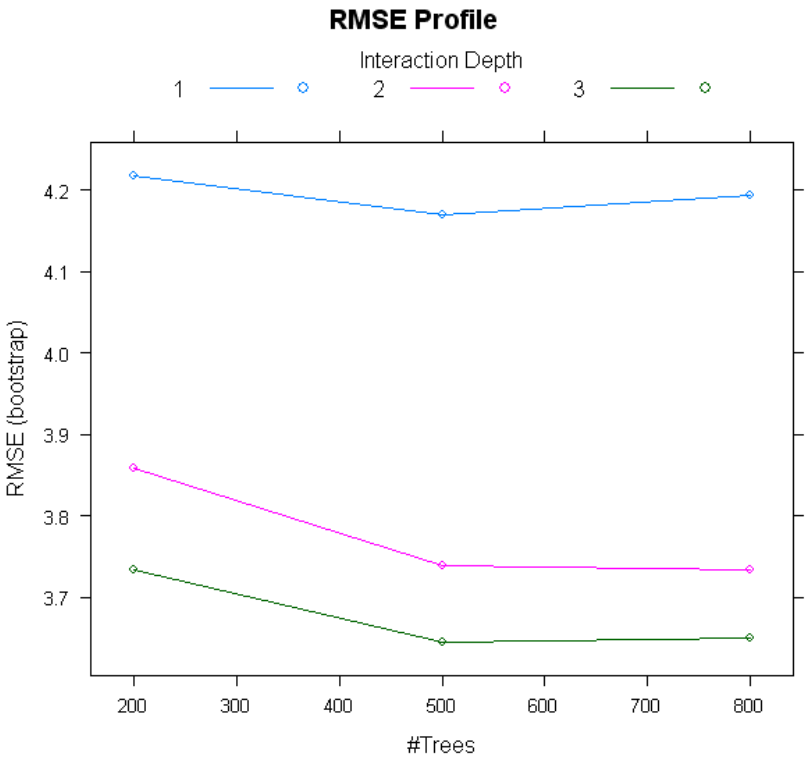
Properties of Boosting

- Robust to overfitting
  - As the number of iterations increases, the test set error does not increase
  - Schapire, et al. (1998), Friedman, et al. (2000), Freund, et al. (2001)
- Can be misled by noise in the response
  - Boosting will be unable to find a predictive model if the response is too noisy.
  - Kriegar, et al. (2002), Wyner (2002), Schapire (2002), Optiz and Maclin (1999)

18

Boosting Trees

- One approach to training is to set the learning rate to a high value (0.1) and tune the other two parameters
- In the plot to the right, a grid of 9 combinations of the 2 tuning parameters were used to optimize the model
- The optimal settings were:
  - 500 trees with high complexity



19

Comparison Summary

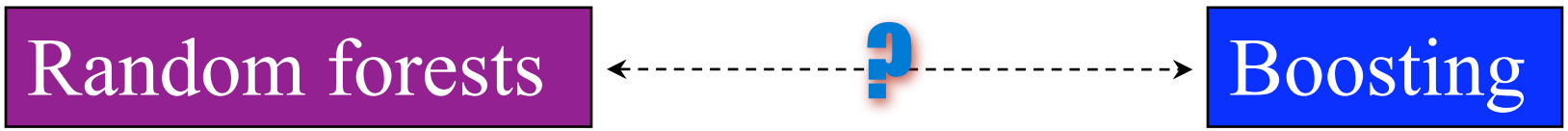
- Comparing the four methods:

	Training Data (bootstrap)		Test	
	RMSE	**Q^2**	**RMSE**	**R^2**
Single Tree	5.18	0.700	4.28	0.780
Bagging	4.32	0.786	3.69	0.825
Rand Forest	3.55	0.857	3.00	0.885
Boosting	3.64	0.847	3.19	0.870

20

Current Research at Pfizer: The best of both worlds?

- Random forests are robust to noise
- Boosting is robust to overfitting
- Can we create a hybrid ensemble that takes advantage of both of these properties?



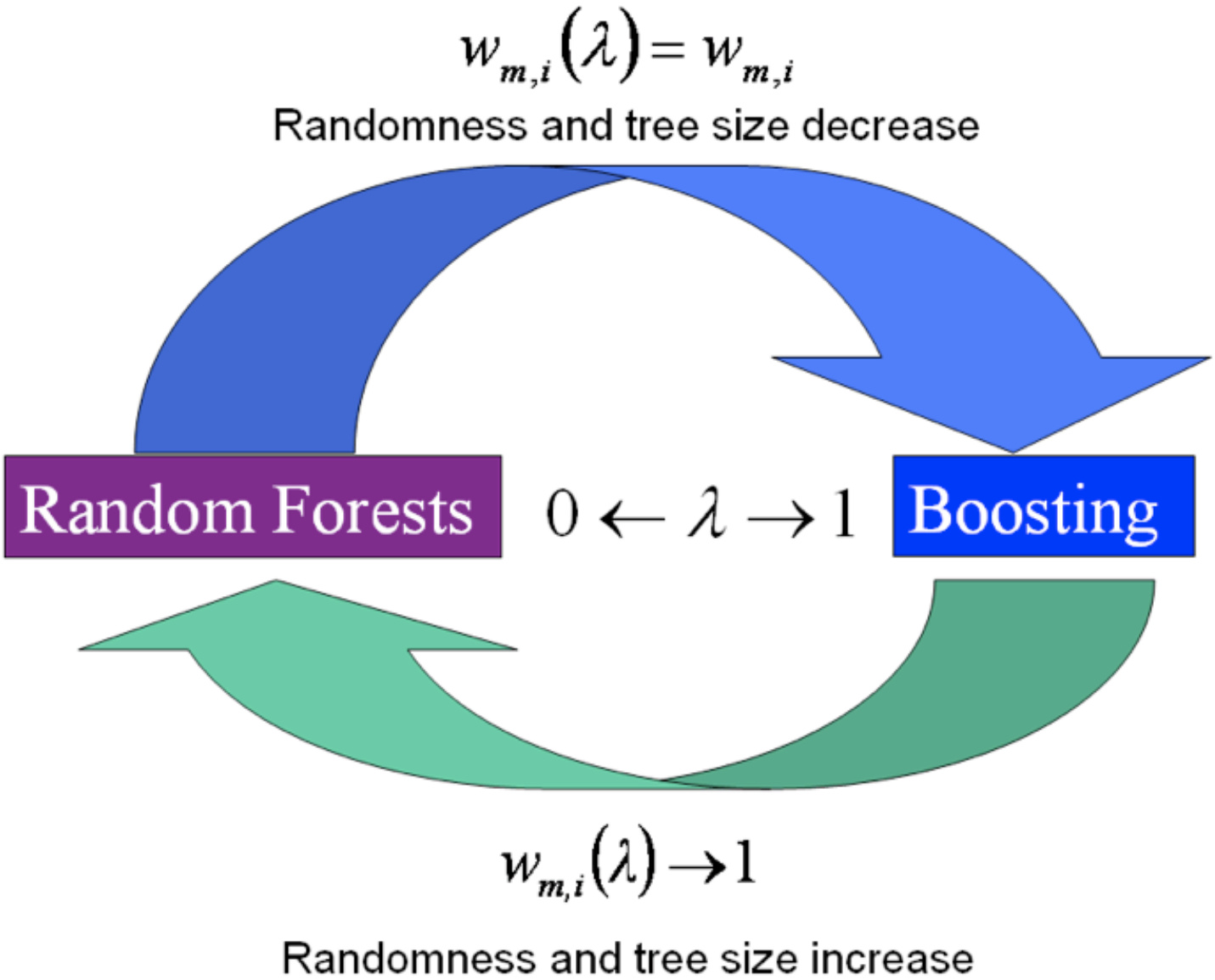
21

Contrasts

- Random forests
  - Prefer large trees
  - Use equally weighted data
  - Use randomness to build the ensemble
- Boosting
  - Prefers small trees
  - Uses unequally weighted data
  - Does not use randomness to build the ensemble
- How to combine these methods?

22

Connecting Random Forests and Boosting



23

Multivariate Adaptive Regression Splines

24

Multivariate Adaptive Regression Splines

- MARS is a nonlinear statistical model
- The model does an exhaustive search across the predictors (and each distinct value of the predictor) to find the best way to sub-divide the data
- Based on this “split” value, MARS creates new features based on that variable
- These artificial features are used to model the outcome

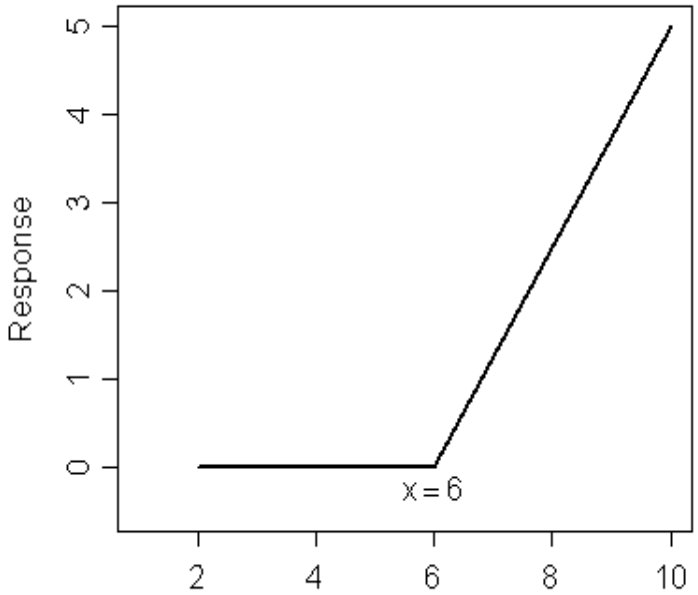
25

MARS Features

- MARS uses “hinge” functions that are two connected lines
- For a data point x of a predictor, MARS creates a function that models the data on each side of x:

$$h(u) = \begin{cases} u & \text{if } u > 0 \\ 0 & \text{otherwise} \end{cases}$$

- These features are created in sets of two (switching which side is “zeroed”)



x	h(x-6)	h(6-x)
2	0	2
4	0	4
8	8	0
10	10	0

26

Prediction Equation and Model Selection

- The model iteratively adds the two new features and uses ordinary regression methods to create a prediction equation. The process then continues iteratively.

- MARS also includes a built-in feature selection routine that can remove model terms
  - the maximum number of retained features (and the feature degree) are the tuning parameters

- The Generalized Cross-Validation statistic (GCV) is used to select the most important terms

$$GCV = penalty \times \sum_{i=1}^{n} \left[y_i - \widehat{f_i(M)} \right]^2$$

$$M = \text{candidate model}$$

$$penalty = \left(1 - \frac{r + 3K}{n}\right)^{-2}$$

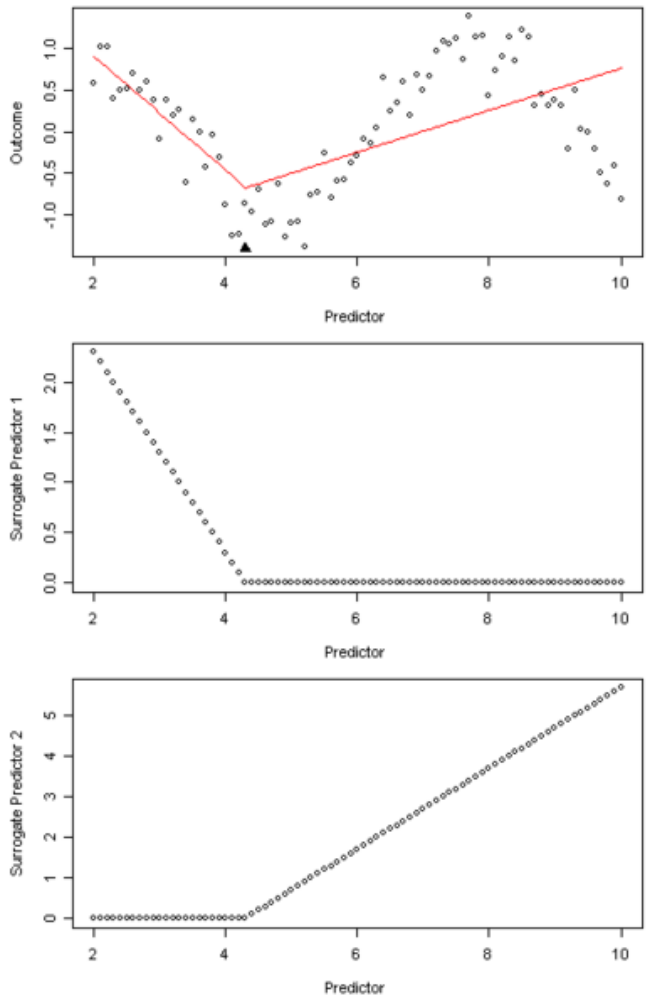
$$r = \text{number of basis functions}$$

$$K = \text{number of knots}$$

27

Sine Wave Example

- As an example, we can use MARS to model one predictor with a sinusoidal pattern
- The first MARS iteration produces a split at 4.3
  - two new features are created
  - a regression model is fit with these features
  - the red line shows the fit

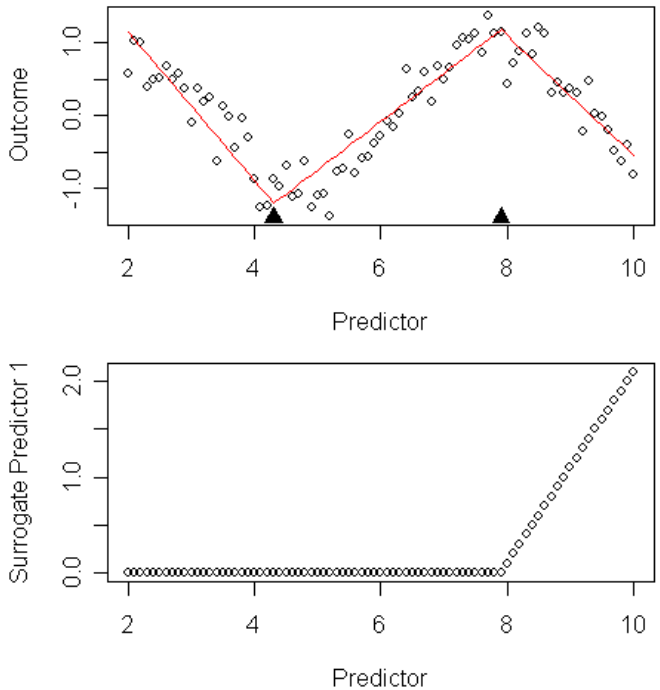


$$\hat{y}_i = \beta_0 + \beta_1\, h(x_i - 4.3) + \beta_2\, h(4.3 - x_i)$$

28

Sine Wave Example

- On the second iteration, a split was found at 7.9
  - two new features are created
- However, the model fit on the left side was already pretty good
  - one of the new surrogate predictors was removed by the automatic feature selection
- The model now has three features

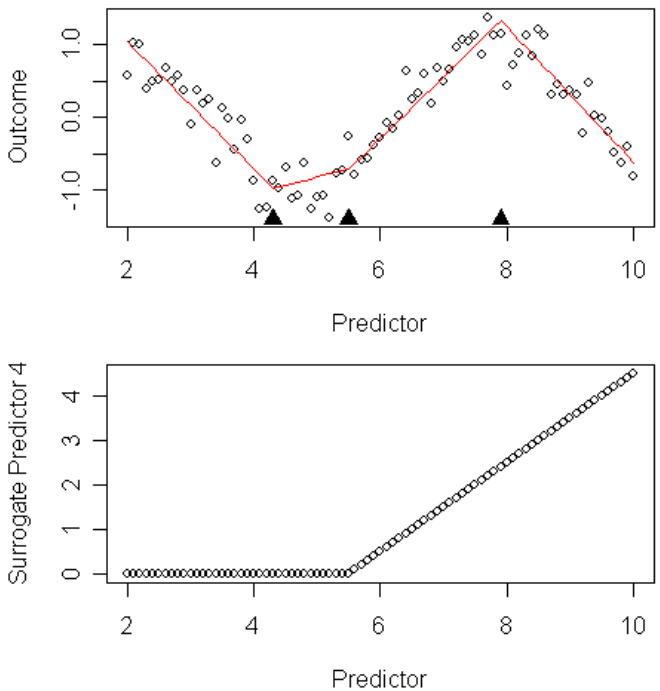


$$\hat{y}_i = \beta_0 + \beta_1\, h(x_i - 4.3) + \beta_2\, h(4.3 - x_i) + \beta_3\, h(x_i - 7.9)$$

29

Sine Wave Example

- The third split occurred at 5.5
- Again, only the “right-hand” feature was retained in the model
- This process would continue until
  - no more important features are found
  - the user-defined limit is achieved

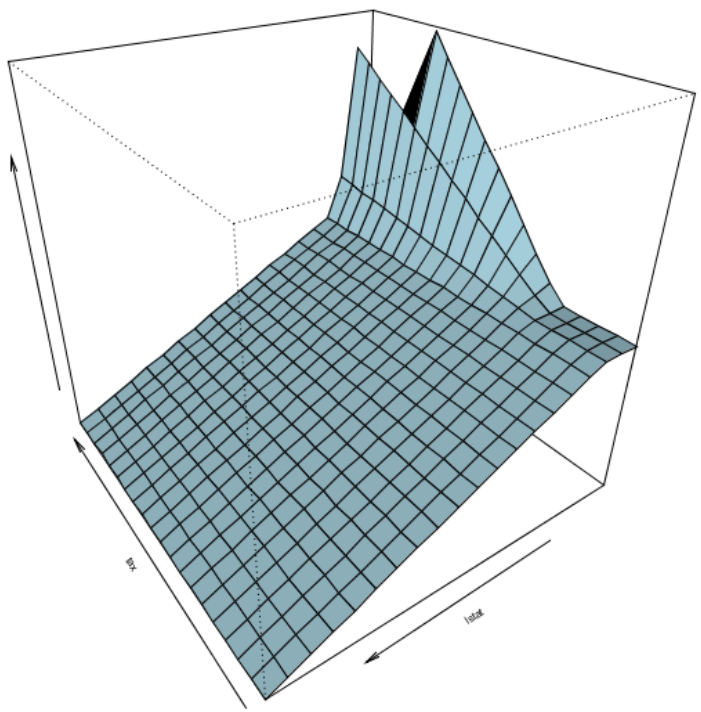


$$\hat{y}_i = \beta_0 + \beta_1\, h(x_i - 4.3) + \beta_2\, h(4.3 - x_i) + \beta_3\, h(x_i - 7.9) + \beta_4\, h(x_i - 5.5)$$

30

Higher Order Features

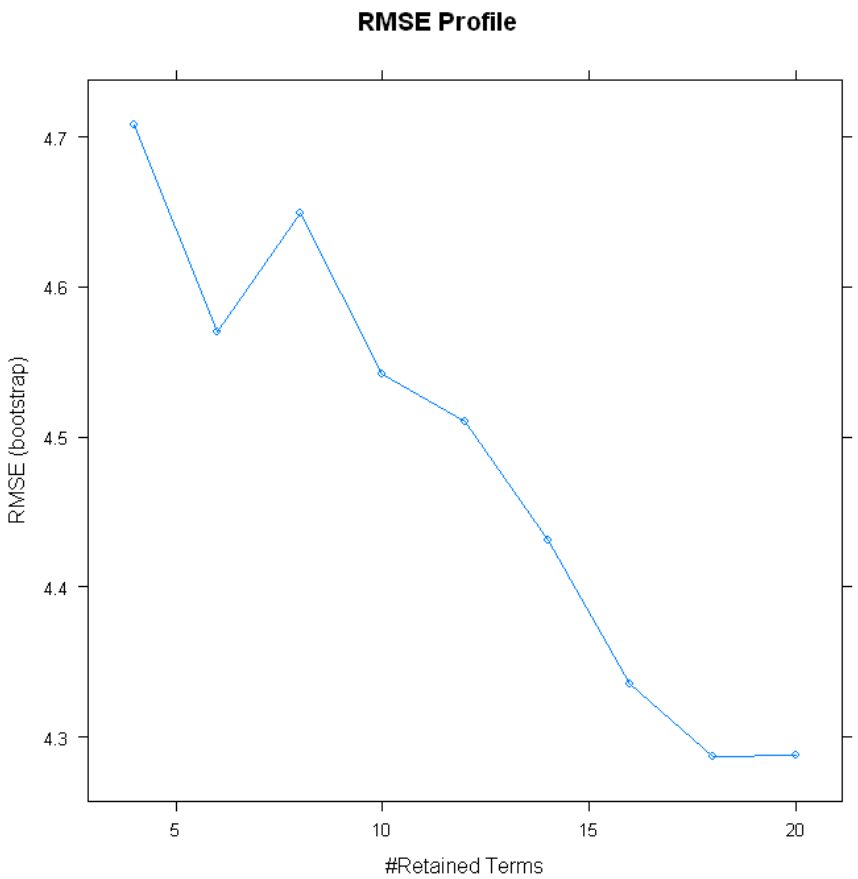
- Higher degree features can also be used
  - two or more hinge functions can be multiplied together to for a new feature
  - in two dimensions, this means that three of four quadrants of the feature can be zero if some features are discarded



31

Boston Housing Data

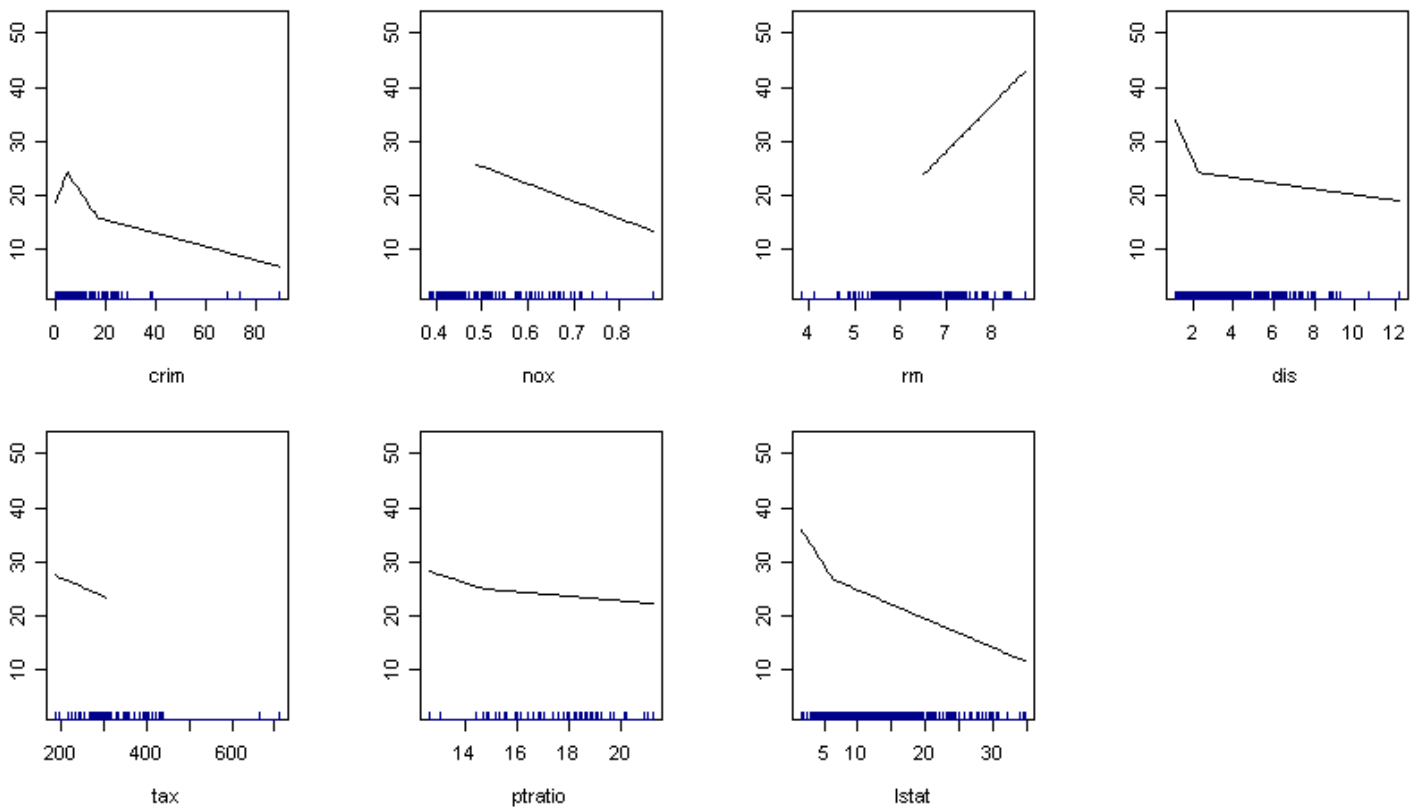
- We tried only additive models
  - the model could retain from 4 to 36 model terms
- The “best” model used 18 terms



32

Boston Housing Data

- Since the model is additive, we can look at the prediction profile of each factor while keeping the others constant



33

Summary

- SVMs are still optimal, but the respectable performance and interpretability of MARS might make us reconsider

	Training Data (bootstrap)		Test Data	
	RMSE	**Q^2**	**RMSE**	**R^2**
Linear Reg	5.23	0.691	4.53	0.742
PLS	5.25	0.689	4.56	0.739
Neural Net	4.60	0.757	4.20	0.780
SVM (radial)	3.79	0.834	3.28	0.861
MARS	4.29	0.791	3.98	0.804

34

Model Building Training

Model Comparisons

35

Which Model is Best?

- The “No Free Lunch Theorem”:
  - over the set of all possible problems, each algorithm will do on average as well as any other

  or, in other words,

  - if one model is better than another, it is because of the particular problem at hand; no one method is uniformly best

- Despite this statement, the next slide has some (subjective) ratings of models

36

Top Level Comparisons

Model	Speed	Performance	Interpretability	Robustness
Boosted Tree				
Random Forest				
Linear Model				
PLS				
MARS				
Neural Net				
SVM				
RDA				
FDA				
Naïve Bayes				

 Excellent Very Good Average Fair Poor

37

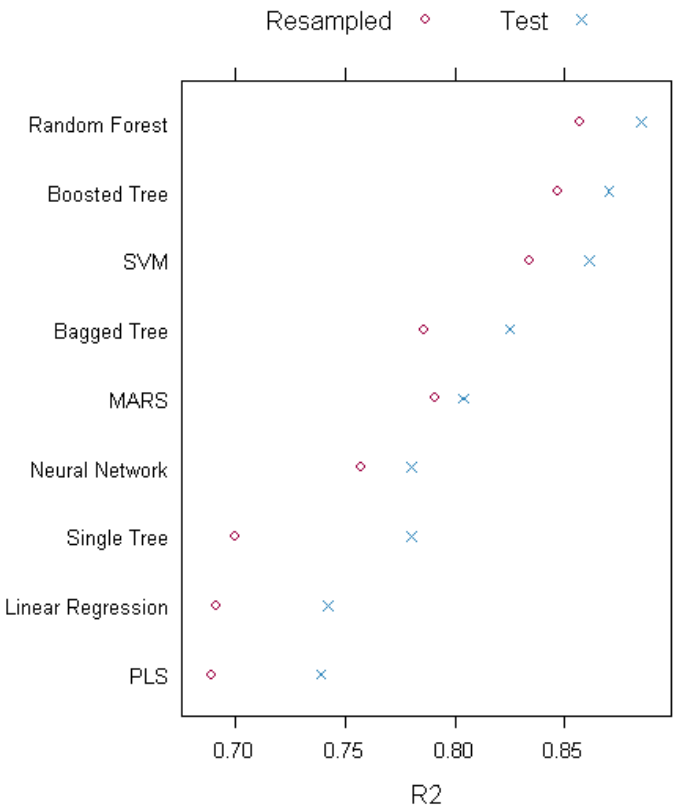
Top Level Comparisons

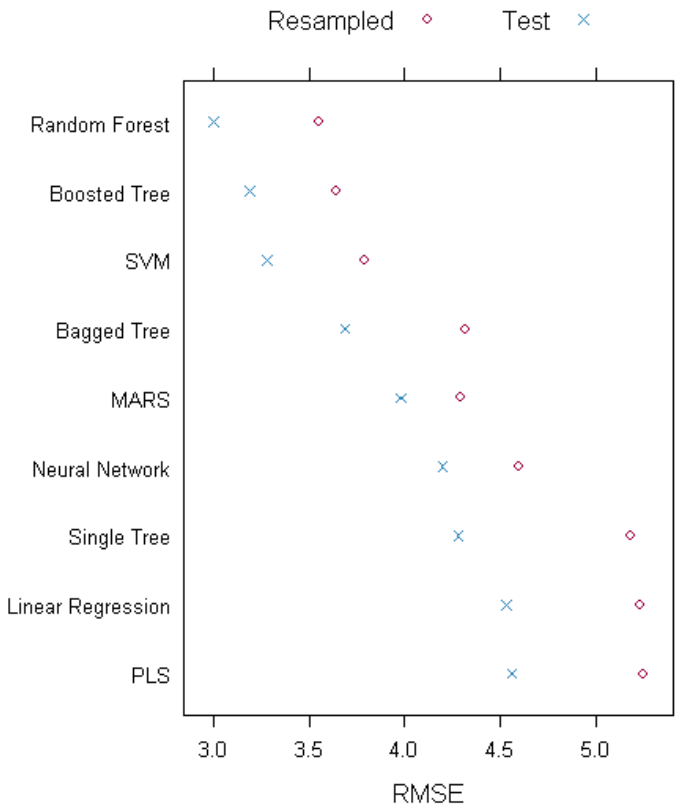
Model	#Param	Pre-Process	P > N ?	Missing Data ?
Boosted Tree	2-3	None	Yes	Yes*
Random Forest	0-1	None	Yes	Yes*
Linear Model	0	ZV, NZV, HCP	No	No
PLS	1	CS	Yes	No
MARS	2	ZV, NZV, HCP	Yes	Yes
Neural Net	2	ZV, CS, HCP	Yes	No
SVM	2-3	CS	Yes	No
RDA	2	ZV	No	No
FDA	2	None	Yes	Yes
Naïve Bayes	0-1	ZV	Yes	Yes

ZV = zero var predictor, NZV = near-zero var predictor,
CS = center+scale, HCP = highly correlated predictor
* Depends on implementation

38

Boston Housing Data



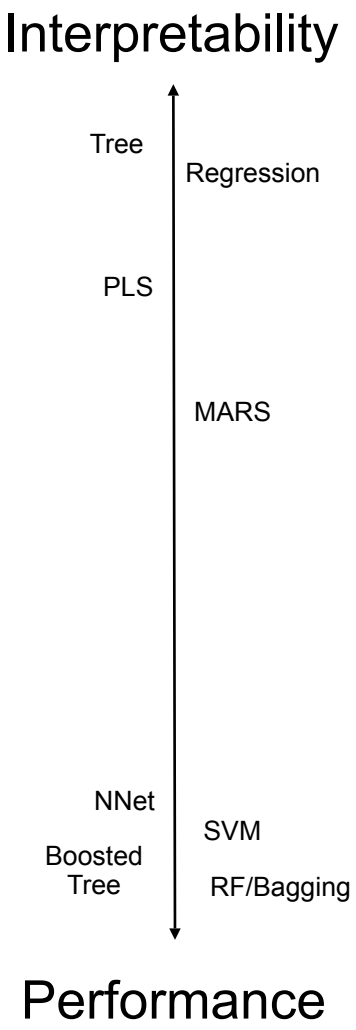


- The correlation between the results on the training set (n=337) via cross-validation and the results from the test set (n=169) were 0.971 (RMSE) and 0.965 (R^2)

39

Some Advice

- There is an inverse relationship between performance and interpretability
- We want the best of both worlds: great performance and a simple, intuitive model
- Try this:
  - Fit a high performance model to get an idea of the best possible performance
  - Move up the line and see if a less complex model can keep performance up with some interpretability



40