RAIS November 20-21, 2025 RESEARCH ASSOCIATION for INTERDISCIPLINARY STUDIES

DOI:10.5281/zenodo.18026535

Nonlinear Regression Modeling via Machine Learning Techniques with Applications in Business and Economics

Sunil K Sapra

Department of Economics and Statistics, California State University, Los Angeles, CA, USA
ssapra@calstatela.edu

Abstract: The paper demonstrates applications of machine learning techniques to economic data. The techniques include nonlinear regression, generalized additive models (GAM), regression trees, bagging, random forest, boosting, and multivariate adaptive regression splines (MARS). Their relative model fitting and forecasting performance are studied. Common algorithms for implementing these techniques and their relative merits and shortcomings are discussed. Performance comparisons among these techniques are carried out via their application to the current population survey (CPS) data on wages and Boston housing data. Overfitting and post-selection inference issues associated with these techniques are also investigated. Our results suggest that the recently developed adaptive machine learning techniques of random forests, boosting, GAM and MARS outperform nonlinear regression model with Gaussian errors and can be scaled to bigger data sets by fitting a rich class of functions almost automatically.

Keywords: Boosting; Generalized Additive Models; Multivariate Adaptive Regression Splines (MARS); Random Forests; Regression Trees; Semi-parametric Regression

JEL Classification Number: C01, C10, C14

Introduction

Linear regression model is the most commonly used method in empirical research in economics and other social sciences. Nevertheless, when the response variable is not continuous, or the functional form and the probability density function of the error term are unknown, linear regression model is not appropriate. Generalized linear models (GLM), extend the linear regression model by allowing the response variable to follow any distribution from the exponential family (for example, normal, Poisson, binomial, gamma etc.). Probit and logit models for discrete choice and Poisson regression for count response are some of the most widely used GLMs in econometric applications. McCullagh and Nelder (1989) provide an authoritative account of GLMs. Cameron and Trivedi (2005) and Greene (2008) provide several econometric applications. Inference for GLMs is based on likelihood theory. Unfortunately, while the GLMs offer flexibility in modeling through a wide choice of link functions, these models limit the choice of the systematic component to a linear combination of predictors and the choice of probability distribution of the response variable to the exponential family, which are likely to fail in many common applications. This calls for the use of nonparametric and nonlinear approaches. A generalized additive model (GAM) is a semi-parametric GLM in which part of the linear predictor is specified in terms of a sum of unknown smooth functions of explanatory variables. GAM is a powerful generalization of linear, logistic, and Poisson regression models. GAMs are very flexible and can provide an excellent fit in the presence of nonlinear relationships. GLMs emphasize estimation and inference for the parameters of the model, while GAMs focus on exploring data nonparametrically and offer more flexibility in model form than the GLMs do. The key benefit of GAMs is that each of the individual additive terms is estimated using a univariate smoother instead of a multivariate smoother for a high-dimensional non-parametric term circumventing the curse of dimensionality: the slow convergence of an estimator to the true value in high dimensions. The GAM formulation of the regression model allows us to build a regression surface as a sum of lower-dimensional nonparametric terms. Regression trees provide an extremely flexible representation of the relationship between the response and the explanatory variables. Unlike linear regression and GAM, regression trees are fully algorithm-based and nonparametric.

Despite a large volume of literature on semi-parametric and nonparametric extensions of linear regression and GLM, there have been no studies to compare these techniques to linear regression model. This paper attempts to fill this gap by studying the relative performance of linear regression, GAM, and regression trees empirically via application of these techniques to current population survey (CPS) data on wages, education, experience, and hours worked. The paper is organized as follows. Section 2 presents the results of a linear regression of wages on education, experience, and hours worked employing the CPS data. Section 3 introduces the GAM, discusses the backfitting algorithm for fitting the GAM and its application to the CPS data. Section 4 presents the regression trees, the recursive partitioning algorithm for fitting regression trees and its application to the CPS data. Section 5 presents bootstrap aggregation of trees (bagging). Since a regression tree by itself is a weak learner, the following sections 6 and 7 study the improvement in predictive performance by combining a large number of deep trees or shallow trees, Section 6 presents random forests based on averaging a large number of deeply grown trees and section 7 presents boosting based on averaging a large number of shallow trees and shrinkage of regression coefficients. Section 8 presents multivariate adaptive regression splines (MARS) and its application to Boston housing data. Section 9 compares the quality of fit and predictive performance of linear regression, regression trees, random forests, gradient boosting, GAMs, and .section 10 provides some concluding remarks.

Non-linear Regression Model

A small subset of CPS data on labor force status are taken from Hill et al (2010). The dependent variable is *ln(WAGE)* and the explanatory variables are *EDUC*, *EXPER*, and *HRSWK*. The variables are defined as follows.

WAGE = Earnings per hour
EDUC = Years of education
EXPER = Post education years experience
HRSWK = Usual hours worked per week
MARRIED = 1 if married.

Table 1. Summary Statistics

Variable	Obs	Mean	Std Dev	Min	Max
WAGE	1000	20.20122	12.1038	2.03	72.13
EDUC	1000	10.689	2.44013	1	16
EXPER	1000	26.501	12.99041	3	64
HRSWK	1000	39.24	11.44611	0	99

Points in a residuals versus *HRSWK* plot from a linear regression of *ln(WAGE)* on *EDUC*, *EXPER*, and *HRSWK* in Fig. 1 are not randomly dispersed points around the horizontal axis and instead display a pattern suggesting a quadratic term in *HRSWK*; as such a nonlinear regression with a quadratic term in HRSWK was estimated. Results are displayed in Table 2.

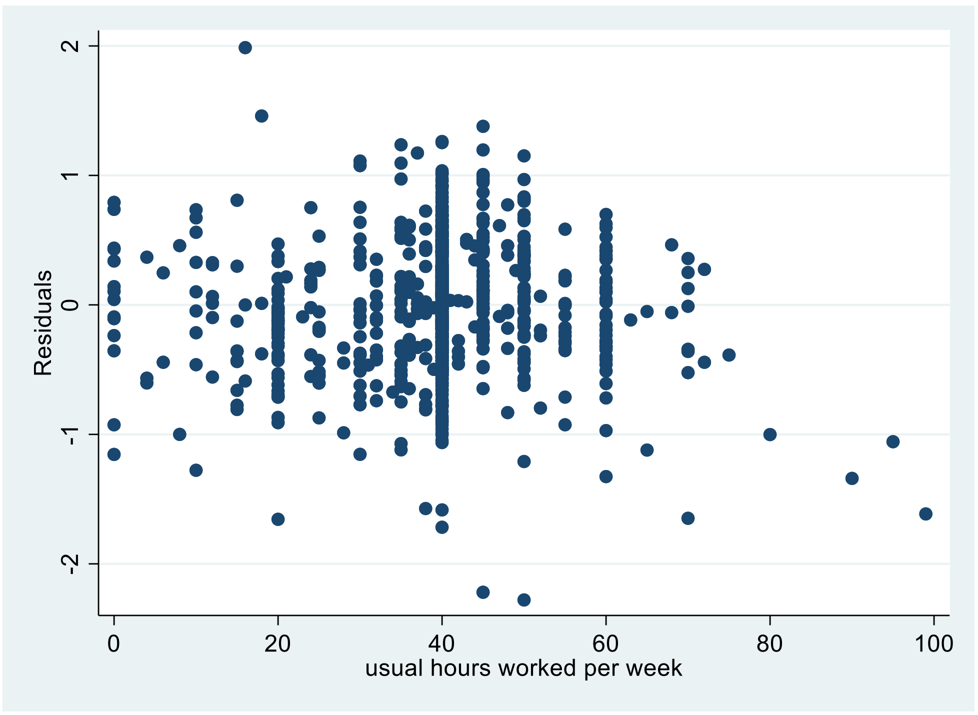


Figure 1. Residuals versus HRSWK plot

A non-linear regression model fitted to the CPS data employs an identity link for *ln(WAGE)* and parametric terms for *EDUC*, *EXPER*, and *HRSWK*:

$$\eta = g(\mu) = \beta_1 + \beta_2 EDUC + \beta_3 EXPER + \quad \beta_3 HRSWK^2 \qquad (1)$$

The results are displayed in Table 2.

Table 2. Non-linear Regression: GLM Normal with Identity link for ln(Wages)

Variable	Coefficient	Std. Error	t-statistic	p-value
INTERCEPT	1.367	0.0841	16.252	< 2e-16 ***
EDUC	0.01039	0.006541	15.881	< 2e-16 ***
EXPER	0.005747	0,001212	4.741	2,43e-06 ***
HRSWK-SQ	0.0001289	0,00001732	7.445	2.09e-13 ***

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4902 on 996 degrees of freedom
Multiple R-squared: 0.2654, Adjusted R-squared: 0.2632
F-statistic: 119.9 on 3 and 996 DF, p-value: < 2.2e-16

Generalized Additive Models (GAM)

GAMs are semi-parametric generalized linear models. These models extend traditional linear models by allowing for a link between the nonlinear nonparametric predictor and the expected value of y. This amounts to allowing for an alternative distribution for the underlying random variation besides just the normal distribution. While Gaussian models can be used in many econometric applications, they are not appropriate for modeling discrete responses such as counts, or bounded responses such as proportions.

GAMs consist of a random component, an additive component, and a link function relating these two components. The response y, the random component, is assumed to have a density in the exponential family

$$f_Y(y\,;\theta\,,\varphi) = exp\left\{\frac{y\theta - b(\theta)}{a(\varphi)} + c(y.\varphi)\right\}, \tag{2}$$

where θ is called the natural parameter and φ is the scale parameter. The normal, binomial, and Poisson distributions are all in this family. Unlike the GLMs, the mean $\mu = E(y \mid x_1, x_2, \ldots, x_p)$ is not linked to the linear predictor $\sum x_{ij}\beta_j$, but to the nonlinear nonparametric predictor

$$\eta = g(\mu) = \alpha + \sum_{j=1}^{p} f_j(x_j), \tag{3}$$

where $f_1(\cdot), \ldots, f_p(\cdot)$ are smooth nonparametric functions, which defines the additive component. Finally, the relationship between the mean μ of the response variable and η is defined by a link function $g(\mu) = \eta$.

A combination of backfitting and local scoring algorithms are used in the actual fitting of the model (Hastie and Tibshirani (1990)). In order to fit GAMs to the data, following Wood (2006), we use basis expansions of smooth functions and penalized likelihood maximization for model estimation in which wiggly models are penalized more heavily than smooth models in a controllable manner, and degree of smoothness is chosen based on cross validation, or AIC or Mallows' criterion.

Estimation of GAM Gaussian Models

The two main approaches to estimation of GAMs are backfitting with local scoring algorithm which employs smoothed partial residuals (Hastie and Tibshirani (1990) and penalized regression splines (PIRLS) (Wood (2006)). Backfitting has the advantage that it can be used with any scatterplot smoother while PIRLS has the advantage that estimation of the smoothing parameter using generalized cross validation is integrated into estimation.

The backfitting algorithm in its simplest form for the Gaussian case is as follows (see Hastie and Tibshirani (1990) and Faraway (2006)).

$$Step\ 1{:}\ Initialize\ by\ setting\ \beta_0 = \bar{y}\ and\ f_j(x) = \hat{\beta}_j x\ where\ \hat{\beta}_j\ is\ an\ intial\ estimate, such\ as\ the\ least\ squares\ for\ j = 1,2,\ldots,p.$$

$$Step\ 2{:}\ Cycle\ j = 1,2,\ldots,p,1,2,\ldots,p,\ldots$$

$$f_j = S(x_j, y - \beta_0 - \sum_{j\neq i} f_j\,(x_i))$$

$where\ S(x,y)\ is\ the\ smooth\ on\ the\ data\ (x,y).$ It could be a nonparametric smoother such as splines or loess or a parametric fit such as linear or polynomial.

The process is iterated until convergence.

Application of GAM to CPS data on wages

GAM Model

The GAM Normal model fitted to the CPS data employs an identity link for *ln(WAGE)* and parametric terms for *EDUC* and *EXPER* but a nonparametric smooth term for *HRSWK*:

$$\eta = g(\mu) = \beta_1 + \beta_2 EDUC + \beta_3 EXPER + f(HRSWK) \tag{4}$$

The results are displayed in Table 3.

Table 3. GAM Normal with Identity link for ln(Wages)

Variable	Coefficient	Std. Error	t-statistic	p-value
INTERCEPT	1.623966	0.079986	20.303	< 2e-16 ***
EDUC	0.101179	0.006287	16.094	< 2e-16 ***
EXPER	0.005256	0.001161	4.527	$6.72\text{x}10^{-6***}$

Approximate significance of smooth terms:

Variable	Estimated df	Refined df	F	p-value
f(HRSWK)	4.266	5.252	28.73	$< 2\text{x}10^{-16}$**

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
R-sq.(adj) = 0.326 Deviance explained = 33%
GCV score = 0.22152 Scale est. = 0.21991 n = 1000
AIC = 1332.581
Number of Local Scoring Iterations: 7

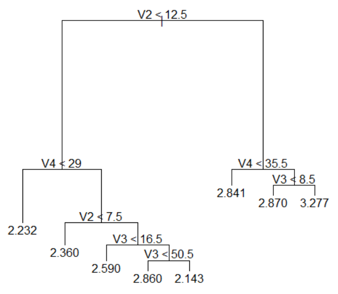
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Regression Trees

Regression trees are algorithm-based and use tree logic to provide predictions. Recursive partitioning is used to grow a large tree as follows. Consider all partitions of the region of the predictors into two regions partitioning each predictor by choosing a point along the range of the predictor to make the split. For each partition, take the mean of the response *y* in the partition to be the predicted value of *y* and compute the residual sum of squares (RSS). Choose the partition that minimizes the RSS. Next, apply cost-complexity pruning to the large tree to obtain a sequence of subtrees as a function of a tuning parameter α. Cross-validation is used to choose the best value of α. The subtree corresponding to the optimal value of α is finally chosen.

Application of Regression Trees to CPS data on wages

In this section, we apply the regression tree to the CPS data on wages used in section 2.



node), split, n, deviance, yval
* denotes terminal node

```
1) root 1000 325.800 2.845
2) V2 < 12.5 664 167.200 2.672
4) V4 < 29 93  19.340 2.232 *
5) V4 > 29 571 126.900 2.744
10) V2 < 7.5 55   6.208 2.360 *
11) V2 > 7.5 516 111.800 2.784
22) V3 < 16.5 123  18.650 2.590 *
23) V3 > 16.5 393  86.980 2.845
46) V3 < 50.5 385  79.510 2.860 *
47) V3 > 50.5 8   3.450 2.143 *
3) V2 > 12.5 336  99.650 3.186
6) V4 < 35.5 45  18.450 2.841 *
7) V4 > 35.5 291  75.010 3.240
14) V3 < 8.5 27   8.927 2.870 *
15) V3 > 8.5 264  62.020 3.277 *
```

Figure 2. Tree model for the CPS data

The first split (node 2) is on V2, 664 observations have V2 (EDUCATION) less than 12.5 with a mean response value of 2.672 while 336 observations have V2 (EDUCATION) greater than 12.5 with a mean response value of 3.186. The total RSS has been reduced from 325.8 to 167.2 + 99.65 = 266.85. The second split (node 2) is on V4 (HRSWK), 93 observations have V4 (HRSWK) less than 29 with a mean response value of 2.232 while 571 observations have V4 greater than 29 with a mean response value of 2.744. The total RSS has been reduced from 266.85 to 19.34 + 126.9 = 146.24. The final split (node 15) is on V3 (EXPERIENCE), 27 observations have V3 (EXPERIENCE) less than 8.5 with a mean response value of 2.870 while 264 observations have V3 greater than 8.5 with a mean response value of 3.277. The total RSS has been reduced from 266.85 to 8.927 + 62.020 = 70.947.

Application of Regression Tree to the Car Seats Data

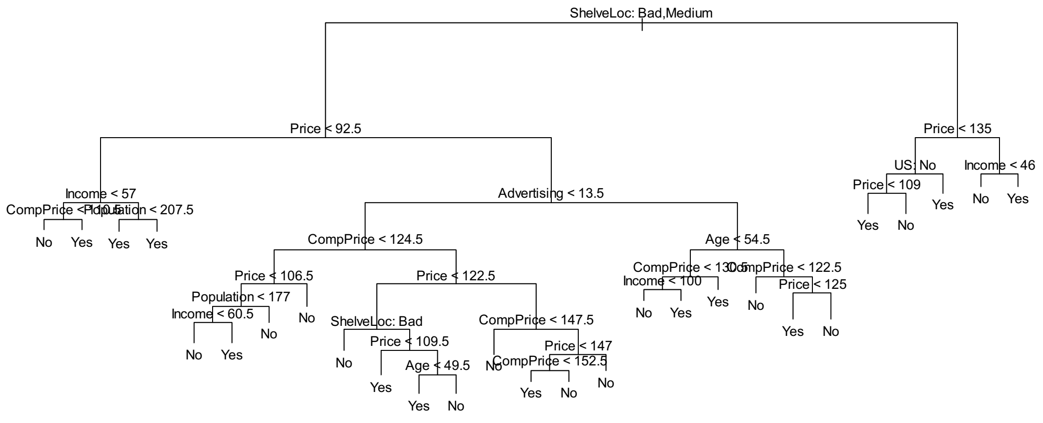


Figure 3. Tree Model for the Boston Housing Data

Bagging

Bagging is a technique designed to reduce the variance of the prediction function f(x) to improve the accuracy of forecasts. The same regression tree is fit many times to bootstrapped versions of the training data and the results are averaged. Each bootstrap tree is likely to have different features than the original tree and a different number of terminal nodes. The bagged estimate is the average predication at x from these B trees.

Suppose we fit a regression tree to our training data $Z = \{(x_1,y_1), (x_2,y_2),\ldots, (x_n,y_n)\}$. Bagging reduces the forecast variance by averaging these predictions over B bootstrap samples. For each bootstrapped training sample Z^{*b}, $b= 1, 2, \ldots, B$, a regression tree is fit to the data, yielding prediction $\widehat{f^{*b}(x)}$. The bagging prediction is then defined by

$$\hat{f}_{bag}(x) = 1/B \sum_{b=1}^{B} \widehat{f^{*b}}(x) \ . \tag{5}$$

In order to study the predictive performance of bagging, we split the CPS data randomly into a training sample consisting of 300 training observations and the test sample consisting of the remaining 700 observations. The root mean squared error of forecasts was 0.4909163. A plot of the predicted values against the test sample observations on V13 (ln(Wages)) also suggests a poor fit illustrating that further improvements in prediction performance may be possible under alternative ensemble methods.

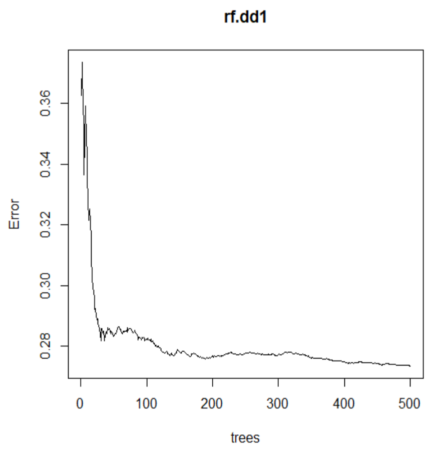
Random Forests

Random forests provide an improvement upon bagging by decorrelating the trees. As in bagging, many deep trees are grown on bootstrapped training samples. However, when constructing these trees, each time a split in a tree is to be made, a random sample of m predictors from the entire set of p predictors is chosen as split candidates. The split can use only one of these m predictors. A new sample of m predictors is taken at each split, typically choosing m ≈ √p predictors at each split.

The main idea underlying random forests is to grow many bushy trees and get rid of the variance through averaging. To benefit from averaging, the technique decorrelates the trees by introducing some randomness in the process of growing trees. This is accomplished as follows. To begin with, the number of trees to be grown/used B and the number of split variables m to be used at each split in a tree are selected. Given the training data set *d = (X,y)*, a bootstrap resampled version of the training data d_b, *b =1,2,…,B* is created by randomly resampling n rows with replacement n times. A maximal depth tree $\widehat{f^{*b}}(x)$ is grown using the data in d_b at each split. Prior to making each split a random subset of $m < p$ features is taken and the search for the best variable is limited to these m of the *p* variables. The random forest fit at any prediction point x is computed as (see Computer-Age Stat Inference)

$$\hat{f}_{rf}(x) = 1/B \sum_{b=1}^{B} \widehat{f^{*b}}(x) \; . \tag{6}$$

Application of Random Forests to CPS data on wages



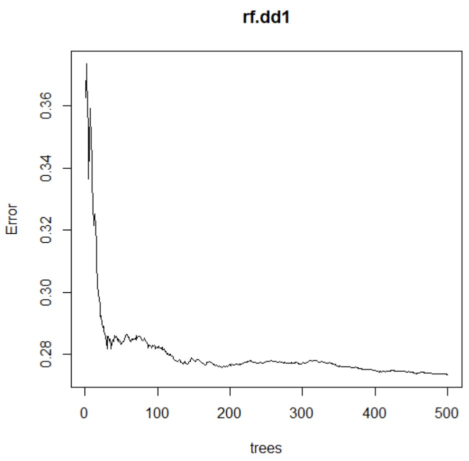


Figure 4. Test Error with variable V2 Figure 5. Test Error with variable V3

Table 4. Variable Importance Measures for a Random Forest with 300 trees

Variable	Mean_min_depth	No. of nodes	MSE_increase	Node_purity_increase	Times_a_root	p-value
V2	1.076667	12299	0.11268457	80.61239	89	1.0
V3	1.013333	16385	0.01719161	53.28146	103	1.173257e-110
V4	1.003333	13881	0.04628184	61.90507	108	9.992388e-01

Application of Random Forests to Boston Housing Data
Housing Values in Suburbs of Boston
Description

The Boston data frame has 506 rows and 14 columns.

This data frame contains the following columns:

Crim: per capita crime rate by town.

Zn: proportion of residential land zoned for lots over 25,000 sq.ft.

Indus: proportion of non-retail business acres per town.

Chas: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).

Nox: nitrogen oxides concentration (parts per 10 million).

Rm: average number of rooms per dwelling.

Age: proportion of owner-occupied units built prior to 1940.

Dis: weighted mean of distances to five Boston employment centres.

Rad: Radon

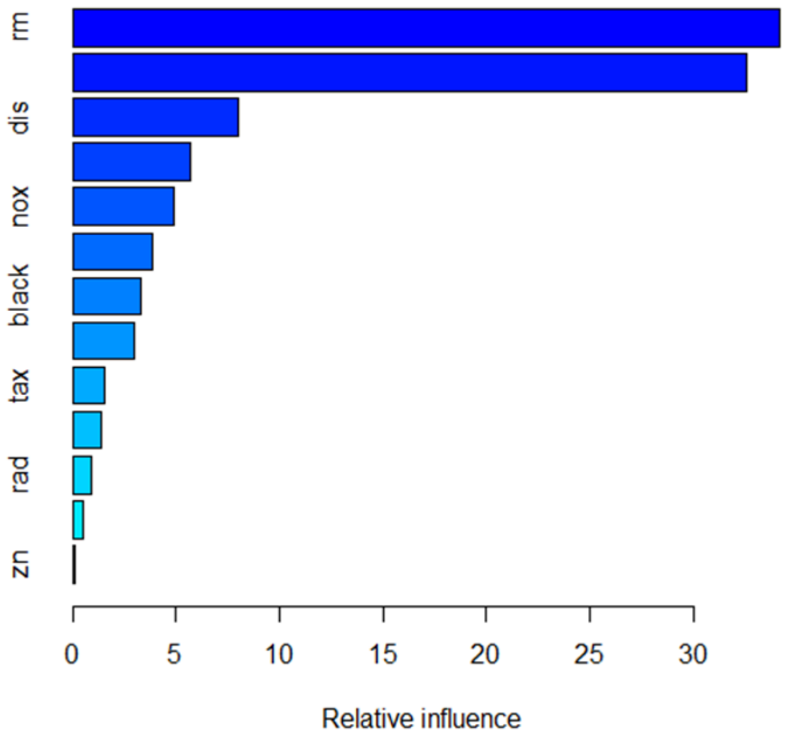


Figure 6. Variable Importance Plot for Boston Housing data

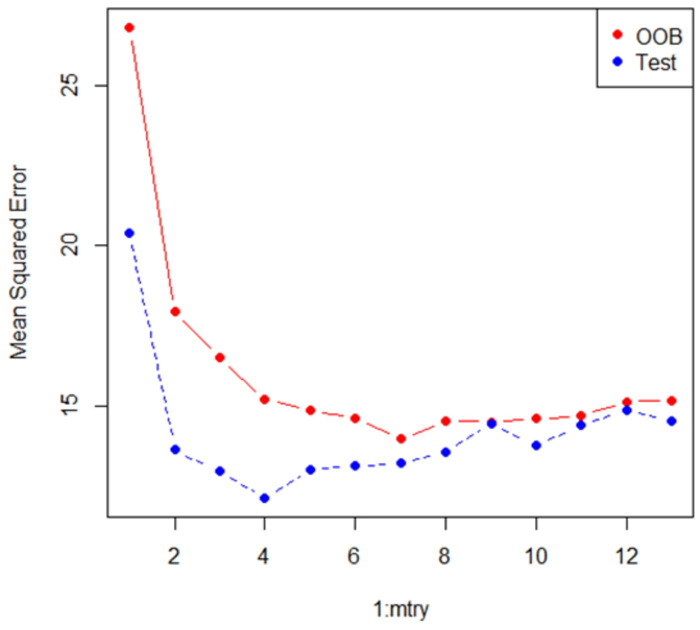


Figure 7. Regression Random Forest with 500 trees with 4 variables at each split

Boosting

Boosting is a general method for building a complex prediction model using simple building components. In its simplest form, regression boosting amounts to the following simple iteration (CASI, p. 320).

1. Initialize b = 0 and $F^0(x) = 0$.
2. For b =1.2.3,…,B:

(a) Compute the residuals $r_i = y_i - F(x_i), i = 1,2, \ldots, , n;$
(b) Fit a small regression tree to the observations $(x_i, r_i), i = 1,2,\ldots,n,$ which can be viewed as estimating a function $g^b(x)$; and update
(c) Update $F^b(x) = F^{b-1}(x) + \epsilon g^b(x).$

The algorithm performs a search in the space of trees for the one most correlated with the residual and then moves the fitted function F^b a small amount in that direction-a process known as forward-stagewise fitting. The number of terms B and the shrinkage parameter ϵ that control the rate of learning (and hence overfitting) are both tuning parameters and are chosen by cross-validation among other methods. Boosting can be viewed as fitting a monotone LASSO path in the high-dimensional space of variables defined by all possible trees of a certain size.

Both bagging and random forests involve fitting many deep trees and fitting the data hard and potentially overfitting. Boosting, in contrast, is a statistical learning approach that learns slowly. Statistical learning approaches that learn slowly tend to perform well by avoiding overfitting.

Under bagging, multiple copies of the original training data are created via the bootstrap, fitting a separate decision tree to each copy/sample, and then combining all of the trees in order to create a single predictive model. Thus, each tree is built on a bootstrap data set, independent of the other trees. While boosting works in a similar fashion, trees are grown sequentially; each tree is grown using information from previously grown trees. Rather than employing bootstrap sampling, each tree is fit on a modified version of the original data set.

Boosting works as follows. Given the current model, the residuals for the model are computed and a decision tree is fit to the residuals from the model rather than the outcome y as the response variable. This new decision tree is added into the fitted function to update the residuals. By fitting small trees with a small tree depth d (or the small number of terminal nodes), the prediction $\hat{f}$ is improved slowly in areas where it does not perform well. The shrinkage parameter slows the process down even more, allowing more and different shapes of trees to update the residuals. Thus, in contrast to bagging, under boosting, construction of each tree depends strongly on the trees that have already been grown. (see ISLR).

A basic version of the popular gradient boosting method is as follows. Given the training data set ***d* = (X,y)**, fix the number of steps B, the tree depth d, and the shrinkage factor ϵ. Set the initial fit $\hat{G}_0 = 0$ and the residual vector $r = y$. For $b = 1, 2, \ldots, B$, fit a regression tree $\widehat{f^{*b}}$ to the data (X,r) grown best-first to depth d. Update the fitted model with a shrunken version of $\widehat{f^{*b}}$: $\hat{G}_b = \hat{G}_{b-1} + \widehat{f^{*bs}}$ with $\widehat{f^{*bs}} = \epsilon f^{*b}$, where $\epsilon > 0$ is a shrinkage factor. Then update the residuals: $r_i = r_i - f^{*bs}$. Finally, output the boosted model $\hat{f}_{boost}(x) = \sum_{b=1}^{B} \widehat{f^{*bs}}(x)$.

Table 5. Relative Influence of Predictors

VARIABLE	RELATIVE INFLUENCE
V3	49.00706
V2	26.60503
V4	24.38791

FIRST TREE IN BOOSTING

```
SplitVar SplitCodePred LeftNode RightNode MissingNode ErrorReduction Weight   Prediction
0       0  1.750000e+01    1      8        12     26.110574    150  8.852991e-05
1       0  9.250000e+00    2      3         7      6.645818     80 -3.814183e-03
2      -1 -9.578651e-03   -1     -1        -1      0.000000     16 -9.578651e-03
3       0  1.311500e+01    4      5         6      1.499012     64 -2.373066e-03
4      -1 -4.284568e-03   -1     -1        -1      0.000000     25 -4.284568e-03
5      -1 -1.147744e-03   -1     -1        -1      0.000000     39 -1.147744e-03
6      -1 -2.373066e-03   -1     -1        -1      0.000000     64 -2.373066e-03
7      -1 -3.814183e-03   -1     -1        -1      0.000000     80 -3.814183e-03
8       0  2.848000e+01    9     10        11      3.698070     70  4.548773e-03
9      -1  2.781926e-03   -1     -1        -1      0.000000     44  2.781926e-03
10     -1  7.538822e-03   -1     -1        -1      0.000000     26  7.538822e-03
11     -1  4.548773e-03   -1     -1        -1      0.000000     70  4.548773e-03
12     -1  8.852991e-05   -1     -1        -1      0.000000    150  8.852991e-05
```

LAST TREE IN BOOSTING

```
SplitVar SplitCodePred LeftNode RightNode MissingNode ErrorReduction Weight   Prediction
0       3  3.100000e+01    1      2        12  0.0033045438    150  1.358195e-05
1      -1  1.368269e-04   -1     -1        -1  0.0000000000     19  1.368269e-04
2       0  8.125000e+00    3      4        11  0.0031545993    131 -4.293264e-06
3      -1 -1.749915e-04   -1     -1        -1  0.0000000000     10 -1.749915e-04
4       0  3.933000e+01    5      9        10  0.0015627850    121  9.814032e-06
5       0  3.339000e+01    6      7         8  0.0001053598    111 -9.728381e-07
6      -1  2.092760e-06   -1     -1        -1  0.0000000000    101  2.092760e-06
7      -1 -3.193538e-05   -1     -1        -1  0.0000000000     10 -3.193538e-05
8      -1 -9.728381e-07   -1     -1        -1  0.0000000000    111 -9.728381e-07
9      -1  1.295483e-04   -1     -1        -1  0.0000000000     10  1.295483e-04
10     -1  9.814032e-06   -1     -1        -1  0.0000000000    121  9.814032e-06
11     -1 -4.293264e-06   -1     -1        -1  0.0000000000    131 -4.293264e-06
12     -1  1.358195e-05   -1     -1        -1  0.0000000000    150  1.358195e-05
```

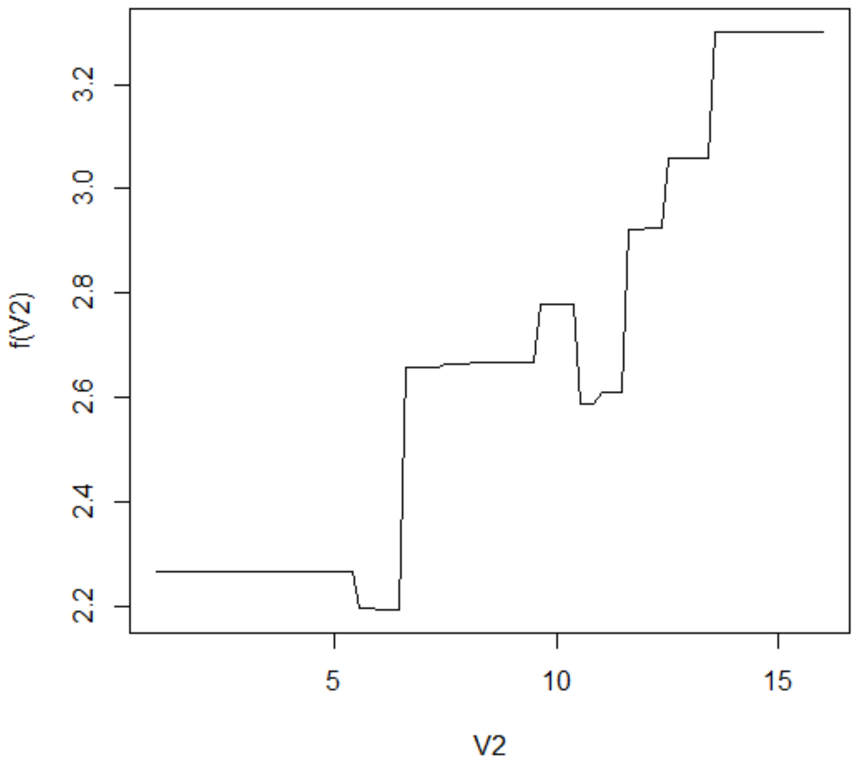


Figure 8

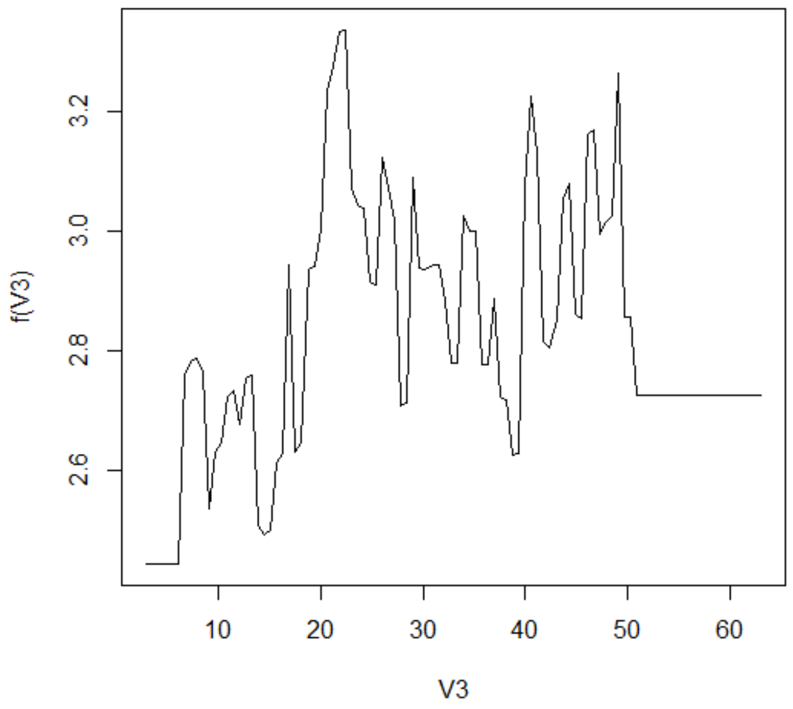


Figure 9

The following plot suggests that OOB underestimates the optimal number of iterations as 215. OOB appears to underestimate the optimal number of iterations (trees employed in the ensemble). OOB generally underestimates the optimal number of iterations although its predictive performance is reasonably competitive. Using cv.folds>0 when calling gbm usually results in improved predictive performance.

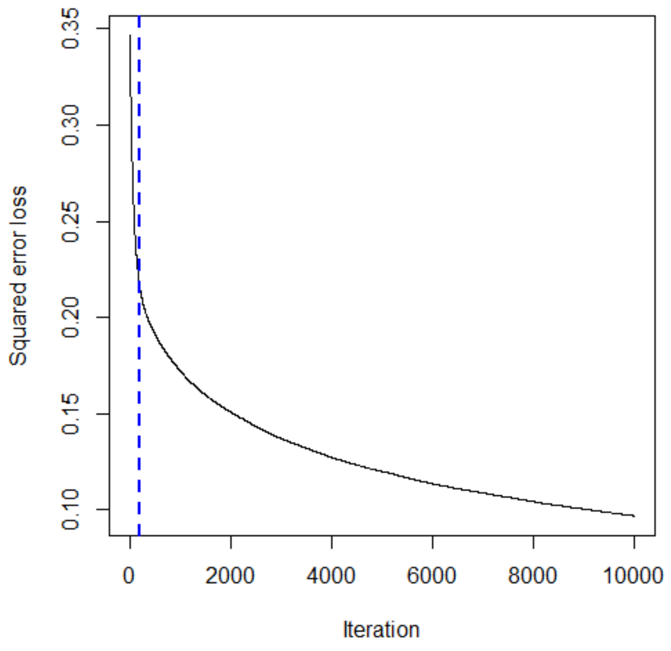


Figure 10. Optimal number of iterations based on OOB error

Therefore, we use cross-validation to determine the optimal number of iterations (trees employed in the ensemble). As the following plot suggests, the optimal number of trees is 9998 (see the dotted vertical line).

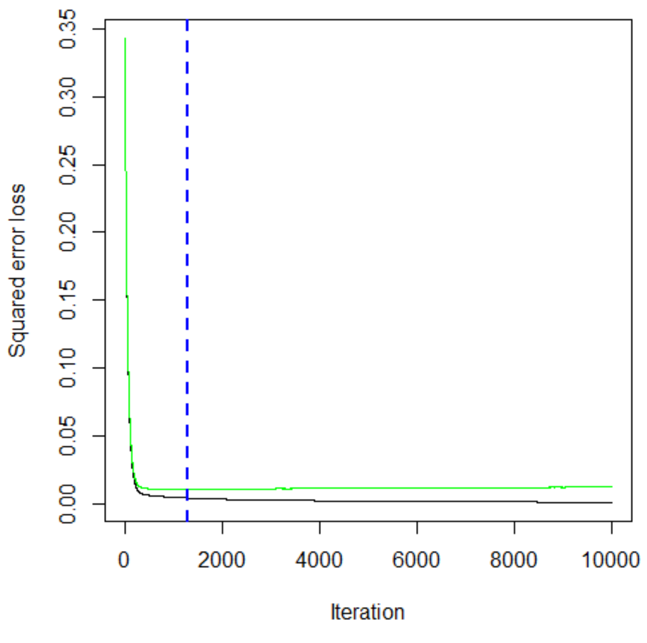


Figure 11. Optimal number of iterations based on 5-fold cross-validation

Table 6. Relative Influence of all 12 Predictors based on the estimated best number of trees.

	var	rel.inf
V1	V1	98.645461285
V4	V4	0.491365557
V3	V3	0.420063987
V2	V2	0.278270801
V5	V5	0.047427113
V7	V7	0.032165944
V6	V6	0.027701503
V9	V9	0.025398773
V8	V8	0.024952126
V10	V10	0.004527560
V11	V11	0.002665352
V12	V12	0.000000000

Next, we plot the marginal effect of the selected variables by "integrating" out the other variables.

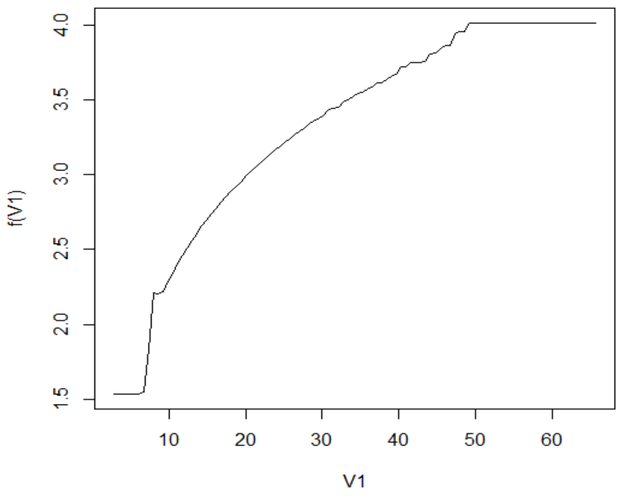


Figure 12

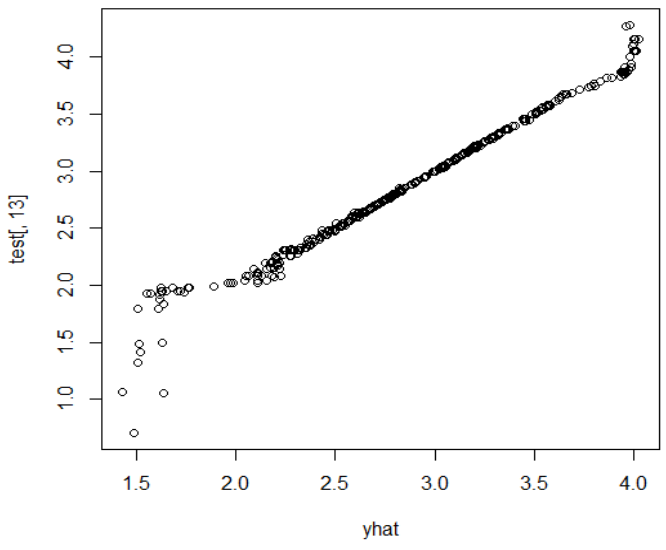


Figure 13

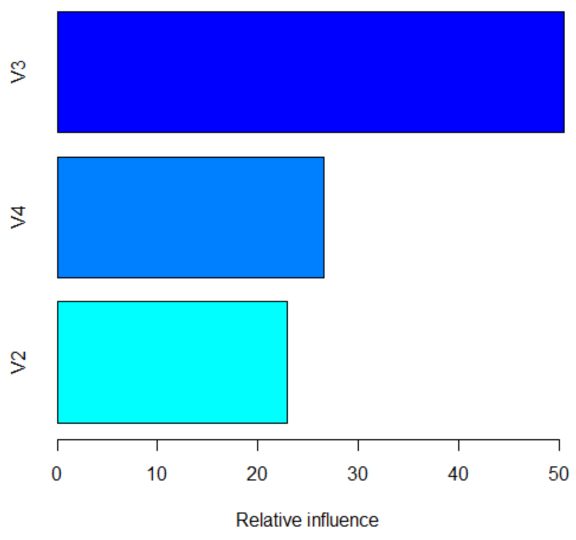


Figure 14. Variable Importance Plot with Gradient Boosting

Multivariate Adaptive Regression Splines (MARS)

Multivariate adaptive regression splines (MARS) is a nonparametric regression technique that automatically models nonlinearities and interactions among predictors.

MARS builds the following kind of model.

$$f(x) = \sum_{\iota=1}^{k} c_\iota B_i(x) \qquad (7)$$

The model is a weighted sum of basis functions $B_i(x)$ and each c_i is a constant coefficient. Each basis function can take one of three forms: a constant 1, a hinge function *max(0, x-constant)* or *max(0, constant-x)*, or a product of two or more hinge functions forming a model interaction between two or more variables.

MARS builds a model in two stages: a forward and a backward pass.

The Forward Pass

1) MARS starts with a model consisting of just the intercept term, which is the average of the values of the response variable y. It then adds basis functions in pairs to the model. At each step, it finds the pair of basis functions, which gives the maximum reduction in residual sum of squares. Each new basis function consists of a term already in the model (which could perhaps be the intercept term) multiplied by a new hinge function. A hinge function is defined by a variable and a knot, so to add a new basis function, MARS must search over all combinations of the following: 1) existing terms (called *parent terms in this context) 2) all variables (to select one for the new basis function) 3) all values of each variable (for the knot of the new hinge function).*

2) To calculate the coefficient of each term MARS applies a linear regression over the terms. This process of adding terms continues until the change in residual error is too small to continue or until the maximum number of terms is reached. The maximum number of terms is specified by the user before model building starts.

The search at each step is done in a brute force fashion, but a key aspect of MARS is that because of the nature of hinge functions the search can be done relatively quickly using a fast least-squares update technique. Actually, the search is not quite brute force. The search can be sped up with a heuristic that reduces the number of parent terms to consider at each step ("Fast MARS")

The Backward Pass

The forward pass usually builds an overfit model. (An overfit model has a good fit to the data used to build the model but will not generalize well to new data.) To build a model with better generalization ability, the backward pass prunes the model. It removes terms one by one, deleting the least effective term at each step until it finds the best submodel. Model subsets are compared using the GCV criterion described below.

The backward pass has an advantage over the forward pass: at any step it can choose any term to delete, whereas the forward pass at each step can only see the next pair of terms.

The forward pass adds terms in pairs, but the backward pass typically discards one side of the pair and so terms are often not seen in pairs in the final model. A paired hinge can be seen in the equation for in the first MARS example above; there are no complete pairs retained in the ozone example.

Generalized Cross-Validation

The backward pass uses generalized cross validation (GCV) to compare the performance of model subsets in order to choose the best subset: lower values of GCV are better. The GCV is a form of regularization: it trades off goodness-of-fit against model complexity.

MARS was applied to the CPS data and the results follow.

Results:

```
summary(earth.mod, digits = 2, style = "pmax")
Call: earth(formula=V13~V2+V3+V4, data=dd)

V13 =
  2
  +  0.12 * pmax(0, V2 -  6)
  -  0.25 * pmax(0, V2 - 10)
  +  0.35 * pmax(0, V2 - 11)
  -  0.12 * pmax(0, V2 - 13)
  - 0.017 * pmax(0, 23 - V3)
  + 0.089 * pmax(0, V3 - 47)
  -  0.68 * pmax(0, V3 - 50)
  +  0.59 * pmax(0, V3 - 51)
  + 0.022 * pmax(0, V4 - 20)
  - 0.034 * pmax(0, V4 - 50)

Selected 11 of 14 terms, and 3 of 3 predictors
Termination condition: Reached nk 21
Importance: V2, V4, V3
Number of terms at each degree of interaction: 1 10 (additive model)
GCV 0.22    RSS 208    GRSq 0.34    RSq 0.36

> summary(earth.mod)
Call: earth(formula=V13~V2+V3+V4, data=dd)
```

```
            coefficients
(Intercept)  1.96591182
h(V2-6)      0.12315704
h(V2-10)    -0.24926354
h(V2-11)     0.34975620
h(V2-13)    -0.12126518
h(23-V3)    -0.01692604
h(V3-47)     0.08913494
h(V3-50)    -0.67731587
h(V3-51)     0.58948714
h(V4-20)     0.02193491
h(V4-50)    -0.03425945

Selected 11 of 14 terms, and 3 of 3 predictors
Termination condition: Reached nk 21
Importance: V2, V4, V3
Number of terms at each degree of interaction: 1 10 (additive model)
GCV 0.2167821    RSS 207.7728    GRSq 0.3359548    RSq 0.362277
```

Interpretation of MARS results

To begin with, for a single knot, our hinge function is h(V2-6) such that

ln(Wage) = 1.96591182 if V2< = 6.
1.96591182 + 0.12315704(V2-6) if V2 > 6.

Once the first knot has been found, search continues for a second knot, which is found at 10. This results in three linear models for ln(Wage):

ln(Wage) = 1.96591182 if V2< = 6.
1.96591182 + 0.12315704(V2-6) if 6 < V2 (EDUC) < = 10
1.96591182 - 0.24926354(V2-10) if V2 (EDUC) > 10,

and so on until the last two knots are found at V4 (HRSWK) = 20 and 50, resulting in the following linear models for the second last knot:

ln(Wage) = 1.96591182 if V4< = 20.
1.96591182 + 0.02193491 (V4-20) if V4 (HRSWK) > 20

And the following linear model for the last knot at V2 = 50:

ln(Wage) = 1.96591182 if V4 < = 20.
1.96591182 + 0.02193491 (V4-20) if 20 < V4 (HRSWK) < = 50
1.96591182 -0.03425945 (V4-50) if V4 (HRSWK) > 50,

The function plot(earth.mod) provides the following comprehensive plots.

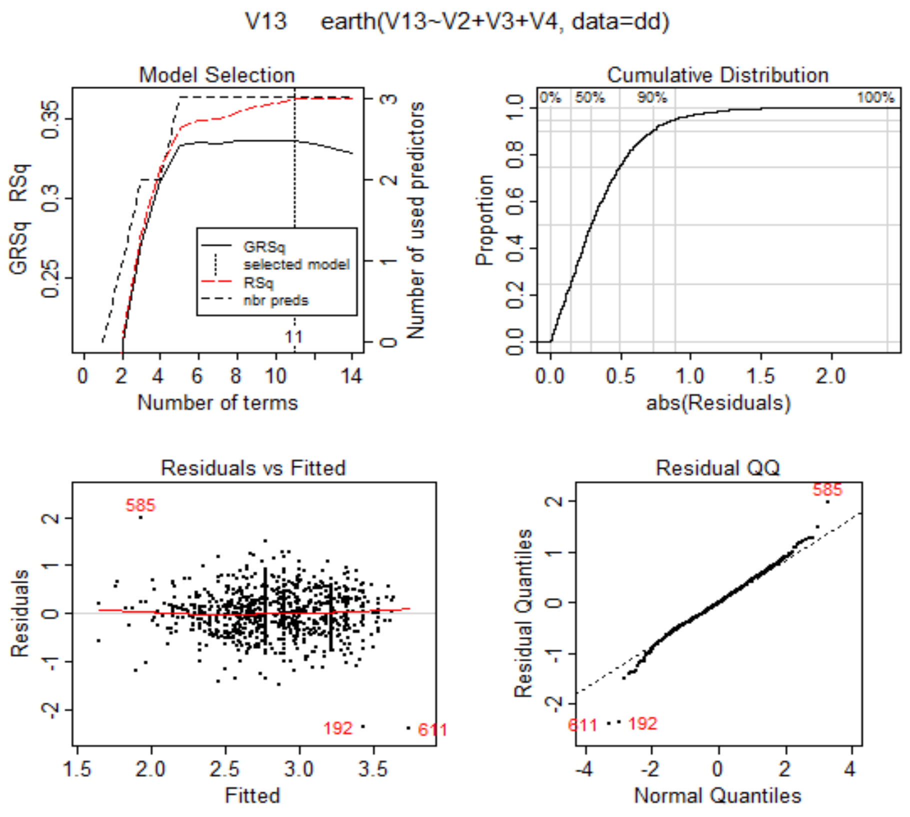


Figure 15. Residual plots with MARS

The function plotmo plots the model's response when varying one or two predictors, See the following plots.

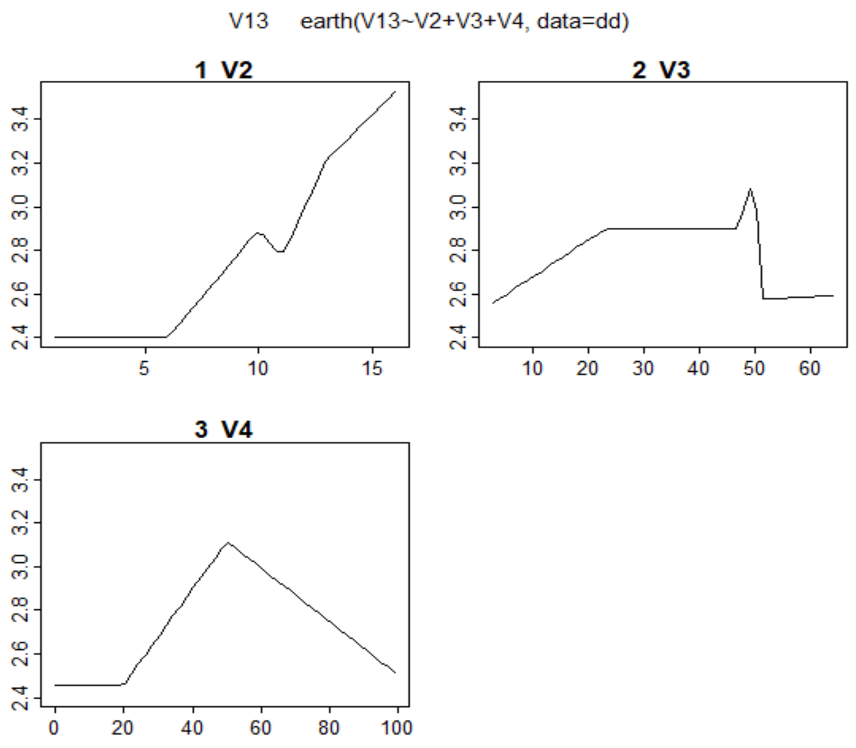


Figure 16. Response variable plots under MARS

We now plot a variance model using the following commands.

```
earth.mod <- earth(V13~V2+V3+V4, data=dd, nfold=10, ncross=30, varmod.method="lm")
plot(earth.mod$varmod) # Plot a variance model (a varmod object).
```

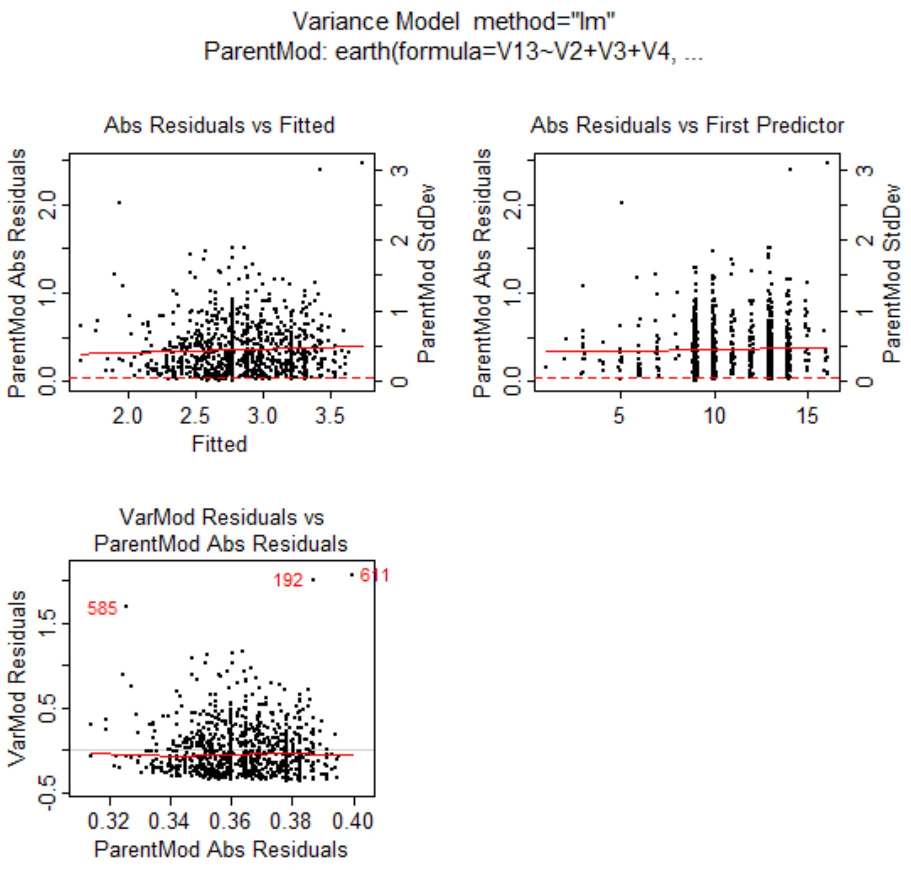


Figure 17

VARIABLE IMPORTANCE USING MARS

ev <- evimp(earth.mod, trim=FALSE) # Estimate variable importances in an earth object plot(ev)

The variable importance plot based on MARS is consistent with the plot obtained using gradient boosting.

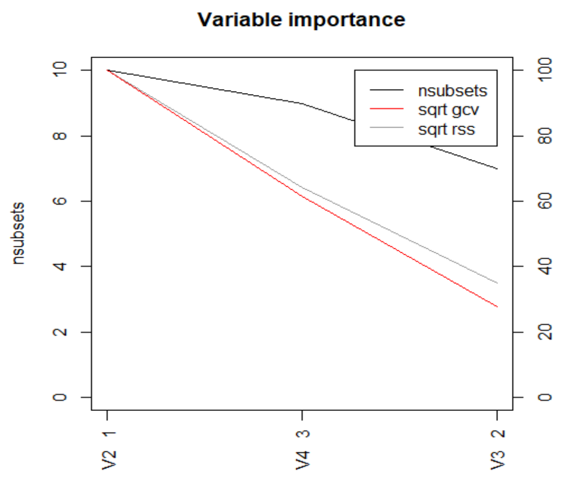


Figure 18. Variable Importance Plot under MARS

```
print(ev)
  nsubsets   gcv    rss
V2      10 100.0  100.0
V4       9  61.5   64.4
V3       7  27.9   34.9
```

Table 7. MARS Regression of ln(WAGES) on EDUC and EXPER

```
Residuals:
    Min       1Q   Median       3Q      Max
-3.02109 -0.30573  0.06603  0.40385  2.13127

Coefficients:

              Estimate  Std. Error  t value    Pr(>|t|)
(Intercept)  1.26863    0.07955    15.948   < 2e-16 ***
a$x[, -1]1   0.29010    0.06793     4.271   2.41e-05 ***
a$x[, -1]2  -0.06034    0.02273    -2.654   0.00825 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6749 on 425 degrees of freedom
Multiple R-squared:  0.1331,    Adjusted R-squared:  0.129
F-statistic: 32.63 on 2 and 425 DF,  p-value: 6.565e-14
GCV = 0.4652906

Selected.terms

0  1 -1
```

The table displays interaction between V12 (EDUC) and V25 (EXPER).

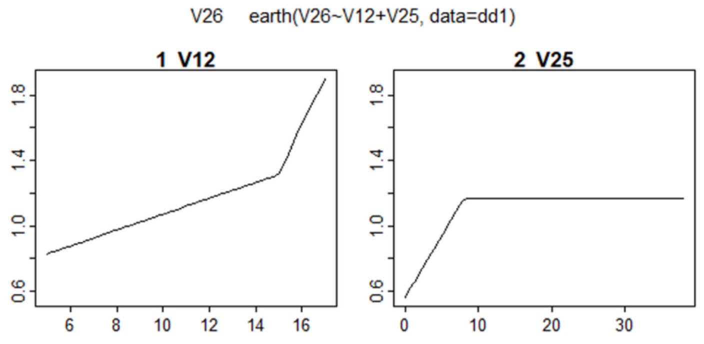


Figure 19

V1= Wage: earnings per hour
V13 = ln(V1) = ln(Wage)
V2 = Educ: years of education
V3 = Exper: post education years experience
V4 = Hrswk: usual hours worked per week

GLM coefficients (family gaussian, link identity)

Table 8. Non-linear Regression: GLM Normal with Identity link for ln(Wages)

Basis Function	Coefficient	Std. Error	t-statistic	p-value
INTERCEPT	3.134224800	0.040630379	77.13994	< 2.22e-16***
h(V2-11)	0.338794345	0.077655453	4.36279	1.4186e-05***
h(11-V2)	-0.084661157	0.013140798	-6.44262	1.8291e-10***
h(V4-50)	-0.010865002	0.003818135	-2.84563	0.00452370**
h(50-V4)	-0.017458842	0.001632114	-10.69707	< 2.22e-16***
h(23-V3)	-0.017496622	0.002577654	-6.78781	1.9576e-11***
*h(V2-14) *h(V3-23)*	-0.019203989	0.005725414	-3.35417	0.00082621***
h(V2-10)	-0.146396050	0.059562451	-2.45786	0.01414712*
h(V3-50)	-0.039759570	0.015452422	-2.57303	0.01022570*

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
GLM dispersion parameter for gaussian family taken to be 0.2125831
Number of cases: 1000
Earth selected 9 of 19 terms, and 3 of 3 predictors
Termination condition: Reached nk 21
Importance: V2, V4, V3
Number of terms at each degree of interaction: 1 7 1
Earth GCV 0.2198047 RSS 210.6698 GRSq 0.326696 RSq 0.3533853
GLM null.deviance 325.8042 (999 dof) deviance 210.6698 (991 dof) iters 2

The basis function of V2 (EDUC) is generated as f(V2) = 0.338794345$(V2-11)_+$ - 0.084661157$(11-V2)_+$, which means that for V2 (EDUC) greater than 11 years, increasing the years of education by 1 year increases wages by approximately 33.88% and for V2 less than 11, increasing years of education might decrease wages by 8.466%, other things equal. The basis function of V4 (HRSWK). is generated as f(V4) = -0.010865002 $(V4-50)_+$ -0.017458842 $(50-V4)_+$, which means that for V4 (HRSWK) greater than 50 hours, increasing the number of hours worked by 1 hour decreases wages by approximately 1.087% and for V4 (HRSWK) less than 50 hours, increasing the number of hours worked by 1 hour decreases wages by approximately 1.74%, other things equal since the overtime wages may be less than regular wages.

To interpret an interaction term, the main effect variables along with their interaction term should be set together in the basis function. For example, the basis function in the model for the interaction of V2 (EDUC) and V3 (EXPER) can be written as:

0.338794345x$(V2-11)_+$ -0.084661157x$(11-V2)_+$ -0.019203989x$(V2-14)_+$*$(V3-23)_+$ Thus, the final coefficient of V2 (EDUC) for V2 (EDUC) greater than 14 years and V3 (EXPER) more than 23 years is 0.338794345 - 0.019203989 = 0.319590356, which suggests that an increase of 1 year in EDUC increases wages by 31.95%.

Comparing the Predictive Techniques

Model Assessment and Selection with Linear Regression, GAM and Regression Trees

An ANOVA test was performed to determine the best model between the linear regression and a GAM, which uses a spline function of *HRSWK*, Results displayed in Table 9 suggest that a nonlinear function of *HRSWK* is needed (p-value = 1.2×10^{-7}) and the small p-value also suggests that the GAM in equation (4) is preferred over the linear regression model.

Table 9. Choosing between Non-linear Regression and GAM

MODEL	RESIDUAL DF	RESIDUAL SS	RESIDUAL DF	F	p-value
Non-linear Regression	996.0	239.340	996	119.9	<2,2e-16
GAM	992.73	218.31	3.2659	15.691	$1.2x10^{-7}$

To study the relative quality of fit to the data for the three methods, Table 10 compares the residual deviance and AIC for the three methods, Both the regression tree and the GAM have significantly lower residual deviance relative to the linear regression model due to their ability to capture nonlinearity in the variable *HRSWK* demonstrating their superior fit over linear regression. The Gaussian GAM with a nonparametric smooth term for *HRSWK* is preferred to the linear regression model due to its substantially lower AIC relative to the linear regression model.

Table 10. Model Assessment and Selection

MODEL	RESIDUAL DEVIANCE	AIC
Non-linear Regression	239.340	1411.008
GAM	218.3125	1332.581
Regression Tree	216.549	-------*

*Regression tree is a nonparametric technique and accordingly AIC, which is likelihood-based is not applicable.

Comparing the forecasting performance of all predictive techniques

We created a random training and test set (600,400) split of the 1000 observations, trained each method on the training set and evaluated its performance on the test set. The relative prediction performance of the three methods was studied by comparing their test set RMSEs.

Table 11 presents the test set RMSEs of forecasts based on six techniques: linear regression, GAM, regression tree, random forests, gradient boosting, and MARS. The test set RMSE of the linear regression of *ln(WAGES)* on *EDUC*, *EXPER*, and *HRSWK* is 0.475756, indicating that the model leads to predictions, which are within around 0.475756% of the mean wages. The RMSE for the GAM with a nonparametric smooth term for hours worked is 0.4587328 indicating that the model leads to predictions, which are within around 0.4587328% of the mean wages. In comparison with the GAM, the test set RMSE of the regression tree is 0.4937155. The lowest test set RMSE for GAM forecasts relative to linear regression as well as regression tree forecasts suggests that GAM outperforms these techniques in terms of out-of-sample performance. A possible explanation lies in the semiparametric character of GAM relative to the fully parametric character of linear regression and the fully nonparametric character of regression trees. This finding also suggests that GAMs are less likely to overfit than linear regression and regression trees.

While GAM outperforms both the linear regression and the GAM, ensemble methods, which combine several regression trees to build a strong learning model, outperform the GAM. The test set RMSE for random forests and gradient boosting are even lower at 0.4555046 and 0.09908411 respectively. Random forests are somewhat more automated and suffer a performance hit as a result relative to gradient boosting. Boosting produces a slow learner and therefore achieves a superior performance (See Efron and Hastie (2016)). MARS has a test set RMSE of 0.4899049 and is barely competitive with regression trees and performs much worse than both random forests and gradient boosting.

Table 11. Test Sample Forecast RMSEs

MODEL	TEST SAMPLE FORECAST RMSE
Non-linear Regression	0,475756
GAM	0.4587328
Regression Tree	0.4937155
Random Forests	0.4555046
Gradient Boosting	0.09908411
MARS	0.4899049

Conclusions

The paper compared the model fit and forecasting performance of non-linear regression with the semiparametric technique of GAM, the nonparametric technique of regression trees, ensemble techniques of random forests and gradient boosting, and multivariate regression splines (MARS). Our results suggest that both the GAM and regression trees provide a better fit to the data than non-linear regression due to their ability to capture the nonlinearity in the predictor *HRSWK*. Interestingly, the GAM outperforms both the non-linear regression and regression tree on forecasting performance as reflected in its lowest RMSE of forecasts. The weakest prediction performance of regression trees can be attributed to its fully nonparametric nature. An alternative nonparametric adaptive technique, multivariate adaptive regression splines (MARS) did not perform well possibly due to its complexity and its potential for overfitting. Two ensemble methods, random forests and gradient boosting markedly outperform both parametric and nonparametric techniques studied by cleverly combining and averaging regression trees. Adaptive, non-parametric techniques, such as MARS, bagging, and boosting are versatile, tend to avoid overfitting, and can be extended to business and economic applications with discrete choice or count data and may be competitive with GAMs and regression trees in terms of out-of-sample performance and interpretability. Performance comparison among these techniques for classification is a topic for future research,

Regression trees by themselves are weak learners and are not competitive with GAMs with regard to their prediction performance.. However, random forests, gradient boosting, and multivariate regression splines (MARS) are fully adaptive and nonparametric and provide reliable predictions despite misspecification due to their reliance on algorithmic models. MARS can also detect interactions among predictors unlike the other procedures.

References

Cameron, A. C., & Trivedi, P. (1998). *Microeconometrics*. Cambridge University Press.

Denison D.G.T., Holmes C.C., Mallick B.K., & Smith A.F.M. (2004). *Bayesian Methods for Nonlinear Classification and Regression*. Wiley.

Efron, B. & Hastie, T. (2016). *Computer-age Statistical Inference,* Cambridge University Press.

Faraway, J. J. (2006). *Extending the Linear Model with R*, Chapman and Hall/CRC, Boca Raton.

Friedman, J. H. (1991). Multivariate Adaptive Regression Splines. *The Annals of Statistics*. *19* (1): 1–67.

Friedman, J. H. (1993) *Fast MARS*. Stanford University Department of Statistics, Technical Report 110.

Greene, W. (2008). *Econometric Analysis*. Pearson/Prentice Hall.

Hastie, T., & Tibshirani, R. (1990). *Generalized Additive Models*. Chapman and Hall.

Heping Zhang &Burton H. Singer (2010) *Recursive Partitioning and Applications*, 2nd edition. Springer.

Hill, R.C., Griffiths, W. E., & Lim, G. (2010). *Principles of Econometrics*. John Wiley & Sons.

McCullagh, P., & Nelder, J. (1989). *Generalized Linear Models*. Chapman and Hall.

Wood, S. (2006). *Generalized Additive Models: an introduction with R*. CRC. Boca Raton.