A General Model-Based Variable Importance Metric for Supervised Learning

Brandon M. Greenwell*
Department of XXX, University of XXX
and
Andrew J. McCarthy
Department of YYY, University of WWW
and
Bradley C. Boehmke
Department of ZZZ, University of WWW

September 26, 2017

Abstract

In the era of "big data", it is becoming more of a challenge to not only build state-of-the-art predictive models, but also gain an understanding of what's really going on in the data. For example, it is often of interest to know which, if any, of the predictors in a fitted model are "important". Fortunately, there are a few modern algorithms—like random forests and boosted decision trees—that have a natural way of ranking predictors in terms of some measure of importance or relative influence. Other algorithms—like neural networks and support vector machines—are not capable of doing so and other model-free approaches are generally used to measure each predictors importance. In this paper, we discuss a model-based approach to measuring predictor importance in any supervised learning setting.

Keywords: 3 to 6 keywords, that do not appear in the title

*The authors gratefully acknowledge *please remember to list all relevant funding sources in the unblinded version*

1 Introduction

Complex supervised learning algorithms, like neural networks and support vector machines, are more common than ever in predictive analytics, especially when dealing with large observational databases that don't adhere to the strict assumptions imposed by traditional statistical techniques (e.g., multiple linear regression which typically assumes linearity, homoscedasticity, and normality). However, it can be challenging to understand the results of such complex models and explain them to management. Graphical displays such as variable importance plots (when available) and partial dependence plots (PDPs) offer a simple solution.

PDPs are low-dimensional graphical renderings of the prediction function $\widehat{f}(\boldsymbol{x})$ so that the relationship between the outcome and predictors of interest can be more easily understood. These plots are especially useful in explaining the output from black box models. While PDPs can be constructed for any predictor in a fitted model, variable importance scores are more difficult to define. Some methods—like random forests and other tree-based methods—have a natural way of defining variable importance. Unfortunately, this is not the case for other popular supervised learning algorithms like support vector machines. In this paper, we offer a solution by providing a partial dependence-based variable importance metric that can be used with any supervised learning algorithm.

2 Background

In the age of "big data", we are often confronted with the task of extracting knowledge from large databases. For this task we turn to various statistical learning algorithms which, when tuned correctly, can have state-of-the-art predictive performance. However, having a model that predicts well is only solving part of the problem. It is still necessary to extract information about the relationships uncovered by the learning algorithm. For instance, we often want to know which predictors, if any, are important by assigning some type of variable importance score to each feature. Once a set of "important" features has been identified, the next step would be to summarize the functional relationship between each feature, or subset thereof, and the outcome of interest. However, since most statistical learning algo-

rithms are "black box" models, extracting this information is not always straightforward. Fortunately, some learning algorithms have a natural way of defining variable importance.

2.1 Model-based approaches to variable importance

In a binary decision tree, at each node t, a single predictor is used to partition the data into two homogeneous groups. The chosen predictor is the one that maximizes some measure of improvement $\widehat{i}_t$. The relative importance of predictor x is just the sum of the squared improvements over all internal nodes of the tree for which x was chosen as the partitioning variable; see Breiman et al. (1984) for details. This idea also extends to ensembles of decision trees like boosting and random forest. In ensembles the the improvement score for each predictor is averaged across all the trees in the ensemble. Fortunately, due to the stabilizing effect of averaging, the improvement-based variable importance metric is often more reliable in large ensembles. Random forests offer an additional way to compute variable importance scores. The idea is to use the left over out-of-bag (OOB) data to construct validation-set errors for each tree. Then each predictor is randomly shuffled in the OOB data and the error is computed again. The idea is that if variable X is important, then the validation error will go up when X is perturbed in the OOB data. The difference in the two errors is recorded for the OOB data for each predictor then averaged across all trees in the forest.

In multiple linear regression, the absolute value of the t statistic is commonly used as a measure of variable importance. The same idea also extends to generalized linear models and nonlinear least squares. Multivariate adaptive regression splines (MARS), which were introduced in Friedman (1991), is an automatic and adaptive regression technique which can be seen as a generalization of multiple linear regression and generalized linear models. In the MARS algorithm, the contribution (or variable importance score) for each predictor is determined using a generalized cross-validation (GCV) statistic.

For neural networks, two popular methods for constructing variable importance scores are the Garson algorithm (Garson, 1991), later modified by Goh (1995), and the Olden algorithm (Olden et al., 2004). Both algorithms use the network's connection weights to form the basis of the variable importance scores. The Garson algorithm determines variable

importance by identifying all weighted connections between the nodes of interest. Olden's algorithm, on the other hand, uses the product of the raw input-hidden and hidden-output connection weights between each input and output neuron and sums the product across all hidden neurons. This has been shown to outperform the Garson method in various simulations.

2.2 Filter-based appraoches to variable importance

Filter-based approaches, which are described in Kuhn and Johnson (2013, chap. 18), do not make use of a fitted model to measure variable importance. They also do not take into account the other variables in the model.

For regression problems, a popular approach is to measuring the variable importance of a numeric predictor X is to first fit a flexible nonparametric model between X and the target $\mathcal{Y}$; for example, the locally-weighted polynomial regression (LOWESS) method of Cleveland (1979). From this fit, a pseudo-R^2 measure can be obtained from the resulting residuals and used as a measure of variable importance. For categorical predictors, a different method based on standatd statistical tests (e.g., t-tests and ANOVAs) are employed; Kuhn and Johnson (2013, chap. 18) for details.

For classification problems, an area under the ROC curve (AUC) statistic can be used to quantify predictor importance. The AUC statistic is derived by using the predictor X as input to the ROC curve. If X can perfectly separate the classes of $\mathcal{Y}$, then that is a good indicator that X is important and this is captured in the corresponding AUC statistic. For problems with more than two classes, extensions of the ROC curve or a one-vs-all approach can be used.

2.3 Partial dependence plots

Harrison and Rubinfeld (1978) were among the first to analyze the well-known Boston housing data. One of their goals was to find a housing value equation using data on median home values from $n = 506$ census tracts in the suburbs of Boston from the 1970 census; see Harrison and Rubinfeld (1978, Table IV) for a description of each variable. The data violate many classical assumptions like linearity, normality, and constant variance. Nonetheless,

Harrison and Rubinfeld—using a combination of transformations, significance testing, and grid searches—were able to find a reasonable fitting model ($R^2 = 0.81$). Part of the payoff for there time and efforts was an interpretable prediction equation which is reproduced in Equation (1).

$$\begin{aligned}\log\widehat{(MV)} = {} & 9.76 + 0.0063RM^2 + 8.98 \times 10^{-5} AGE - 0.19 \log(DIS) + 0.096 \log(RAD) \\ & - 4.20 \times 10^{-4} TAX - 0.031 PTRATIO + 0.36 (B - 0.63)^2 - 0.37 \log(LSTAT) \\ & - 0.012 CRIM + 8.03 \times 10^{-5} ZN + 2.41 \times 10^{-4} INDUS + 0.088 CHAS \\ & - 0.0064 NOX^2. \end{aligned} \tag{1}$$

Nowadays, many supervised learning algorithms can fit the data automatically in seconds—typically with higher accuracy. The downfall, however, is some loss of interpretation since these algorithms typically do not produce simple prediction formulas like Equation (1). These models can still provide insight into the data, but it is not in the form of simple equations. For example, quantifying predictor importance has become an essential task in the analysis of "big data", and many supervised learning algorithms, like tree-based methods, can naturally assign variable importance scores to all of the predictors in the training data.

While determining predictor importance is a crucial task in any supervised learning problem, ranking variables is only part of the story and once a subset of "important" features is identified it is often necessary to assess the relationship between them (or subset thereof) and the response. This can be done in many ways, but in machine learning it is often accomplished by constructing *partial dependence plots* (PDPs); see Friedman (2001) for details. PDPs help visualize the relationship between a subset of the features (typically 1-3) and the response while accounting for the average effect of the other predictors in the model. They are particularly effective with black box models like random forests and support vector machines.

Let $\boldsymbol{x} = \{x_1, x_2, \ldots, x_p\}$ represent the predictors in a model whose prediction function is $\widehat{f}(\boldsymbol{x})$. If we partition $\boldsymbol{x}$ into an interest set, $\boldsymbol{z}_s$, and its compliment, $\boldsymbol{z}_c = \boldsymbol{x} \setminus \boldsymbol{z}_s$, then the "partial dependence" of the response on $\boldsymbol{z}_s$ is defined as

$$f_s(\boldsymbol{z}_s) = E_{\boldsymbol{z}_c}\left[\widehat{f}(\boldsymbol{z}_s, \boldsymbol{z}_c)\right] = \int \widehat{f}(\boldsymbol{z}_s, \boldsymbol{z}_c)\, p_c(\boldsymbol{z}_c)\, d\boldsymbol{z}_c, \tag{2}$$

where $p_c(\boldsymbol{z}_c)$ is the marginal probability density of $\boldsymbol{z}_c$: $p_c(\boldsymbol{z}_c) = \int p(\boldsymbol{x})\, d\boldsymbol{z}_s$. Equation (?) can be estimated from a set of training data by

$$\bar{f}_s(\boldsymbol{z}_s) = \frac{1}{n}\sum_{i=1}^{n} \widehat{f}(\boldsymbol{z}_s, \boldsymbol{z}_{i,c}), \tag{3}$$

where $\boldsymbol{z}_{i,c}$ $(i = 1, 2, \ldots, n)$ are the values of $\boldsymbol{z}_c$ that occur in the training sample; that is, we average out the effects of all the other predictors in the model.

Constructing a PDP (3) in practice is rather straightforward. To simplify, let $\boldsymbol{z}_s = x_1$ be the predictor variable of interest with unique values $\{x_{11}, x_{12}, \ldots, x_{1k}\}$. The partial dependence of the response on x_1 can be constructed as follows:

1. For $i \in \{1, 2, \ldots, k\}$:
   (a) Copy the training data and replace the original values of x_1 with the constant x_{1i}.
   (b) Compute the vector of predicted values from the modified copy of the training data.
   (c) Compute the average prediction to obtain $\bar{f}_1(x_{1i})$.
2. Plot the pairs $\{x_{1i}, \bar{f}_1(x_{1i})\}$ for $i = 1, 2, \ldots, k$.

Algorithm 1: A simple algorithm for constructing the partial dependence of the response on a single predictor x_1.

Algorithm 1 can be quite computationally intensive since it involves k passes over the training records. Fortunately, the algorithm can be performed in parallel quite easily (more on this in Section **??**). It can also be easily extended to larger subsets of two or more features as well.

2.4 Boston housing data

For illustration, we will use a corrected version of the Boston housing data analyzed in Harrison and Rubinfeld (1978); the data are available from Statlib at `http://lib.stat.cmu.edu/datasets/boston_corrected.txt`. Using the R package `randomForest` (Liaw

and Wiener, 2002), we fit a random forest with tuning parameter $m_{try} = 6$ (chosen using 5-fold cross-validation) and 1000 trees. The model fit is reasonable, with an *out-of-bag* (pseudo) R^2 of 0.89. The variable importance scores are displayed in Figure **??**. Both plots indicate that the percentage of lower status of the population (`lstat`) and the average number of rooms per dwelling (`rm`) are highly associated with the median value of owner-occupied homes (`cmedv`). They also indicate that the proportion of residential land zoned for lots over 25,000 sq.ft (`zn`) has little association with `cmedv`.

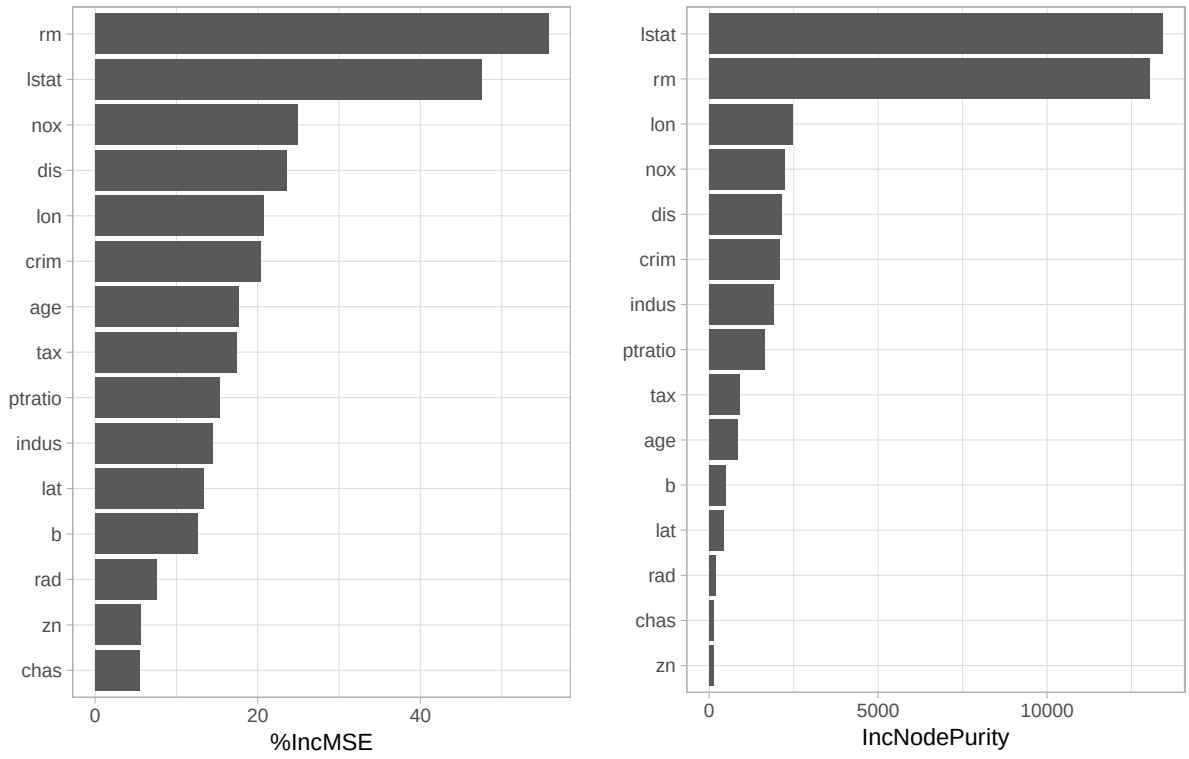

Figure 1: Variable importance scores from a random forest fit to the (corrected) Boston housing data. *Left*: OOB-based variable importance scores. *Right*: Impurity-based variable importance scores.

The partial dependence functions for these three variables are displayed in Figure **??**. Notice how the PDP for `zn` is essentially flat. It is this notion of "flatness" which we will use to define our variable importance measure.

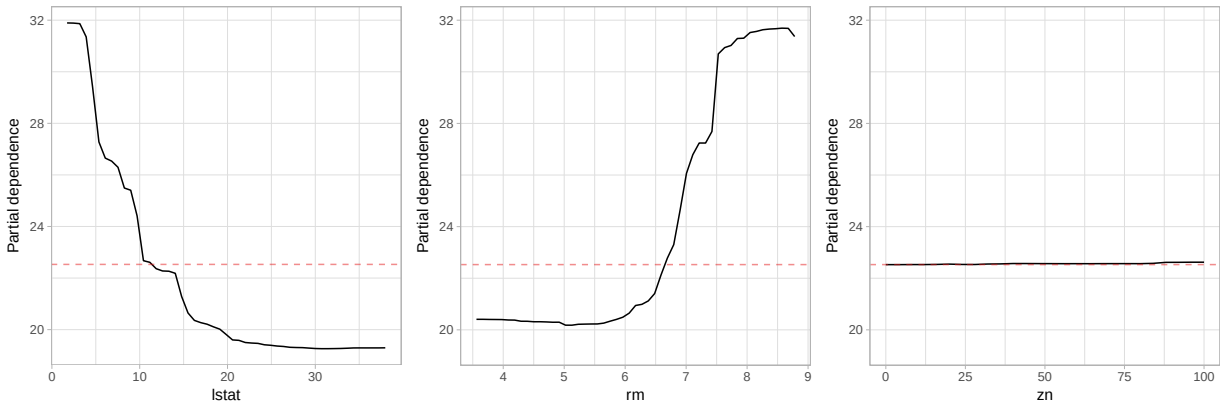

Figure 2: Partial dependence of `cmedv` on `lstat` (left), `rm` (middle), and `zn` (right). The mean of the $n = 506$ response values is indicated by a dashed red line.

3 A model-based variable importance measure

The PDP for `zn` in Figure **??** is relatively flat indicating that `zn` does not have much influence on the predicted median home value. In other words, the values $\bar{f}(x_i)$ display little variability. One might conclude that any variable for which the PDP is "flat" is likely to be less important than those predictors whose PDP varies across a wider range of the response.

Our notion of variable importance is based on any measure of the "flatness" of the partial dependence function. In general, we define

$$\text{imp}(x) = \text{flatness}\left(\bar{f}_s(\boldsymbol{z}_s)\right), \tag{4}$$

where flatness is any measure of "flatness" of the curve. A simple and effective measure to use is the sample standard deviation for continuous predictors and the range statistic for factors with K levels. Based on Algorithm 1, our importance metric for predictor x_1 is simply

$$\text{imp}(x_1) = \sqrt{\frac{1}{k-1}\sum_{i=1}^{k}\left[\bar{f}_1(x_{1i}) - \frac{1}{k}\sum_{i=1}^{k}\bar{f}_1(x_{1i})\right]^2}, \tag{5}$$

if x_1 is continuous and

$$\text{imp}(x_1) = \max_i\left(\bar{f}_1(x_{1i})\right) - \min_i\left(\bar{f}_1(x_{1i})\right), \tag{6}$$

if x_1 is categorical. Note that our variable importance metric relies on the fitted model; hence, it is crucial to properly tune and train the model to have good performance.

To illustrate, we apply the partial dependence algorithm 1 to all of the predictors in the random forest for the Boston housing example and compute (5). The results are displayed in Figure **??**.

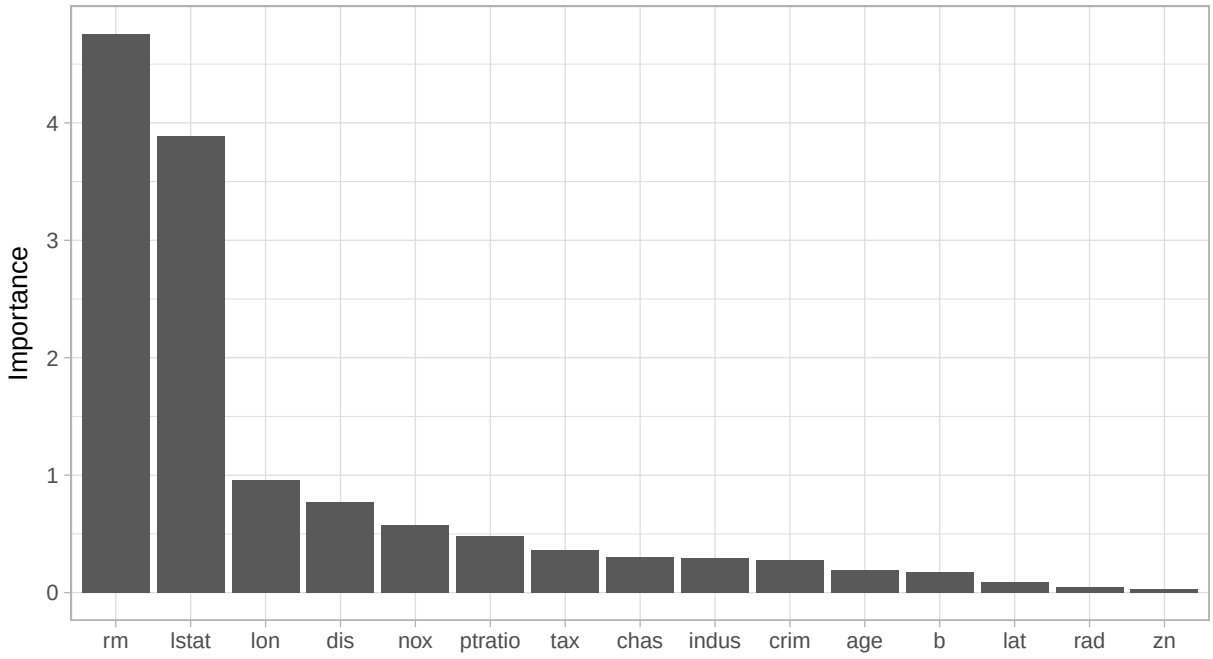

Figure 3: Partial-dependence-based variable importance scores from a random forest fit to the (corrected) Boston housing data.

3.1 Linear models

As mentioned earlier, for linear models, a natural choice for measuring the importance of each term is to use the absolute value of the corresponding coefficient divided by its estimated standard error (i.e., the absolute value of the t-statistic). This turns out to be equivalent to the partial dependence-based metric when the predcitors are independely and uniformly distributed of the same range.

For example, suppose we have a linear model of the form

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon,$$

where β_i $(i = 1, 2)$ is a constant, X_1 and X_2 are both $\mathcal{U}(0, 1)$ random variables, and $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Since we know the distribution of X_1 and X_2, we can easily find $f_1(X_1)$ and $f_2(X_2)$. For instance,

$$f_1(X_1) = \int_0^1 E[Y|X_1, X_2]\, p(X_2)\, dX_2,$$

where $p(X_2) = 1$. Simple calculus leads to

$$f_1(X_1) = \beta_0 + \beta_2/2 + \beta_1 X_1 \quad \textit{and} \quad f_2(X_2) = \beta_0 + \beta_1/2 + \beta_2 X_2..$$

Since $E[Y|X_1, X_2] = f(X_1, X_2)$ is additive, the true partial dependence functions are just simple linear regressions in each predictor with their original coefficient and an adjusted intercept. Taking the variance of each gives

$$Var[f_1(X_1)] = \beta_1^2/12 \quad \textit{and} \quad Var[f_2(X_2)] = \beta_2^2/12.$$

Hence, the standard deviations are just the absolute values of the original coefficients (scaled by the same constant).

To illustrate, we simulated $n = 1000$ observations from the following linear model

$$Y = 1 + 3X_1 - 5X_2 + \epsilon,$$

where X_1 and X_2 are both $\mathcal{U}(0, 1)$ random variables, and $\epsilon \sim \mathcal{N}(0, 0.01^2)$. Since we know the distribution of X_1 and X_2, we can easily find $f_1(X_1)$ and $f_2(X_2)$:

$$f_1(X_1) = \int_0^1 (1 + 2X_1 - 5X_2)\, p(X_2)\, dX_2,$$

where $p(X_2) = 1$. Simple calculus leads to

$$f_1(X_1) = -\frac{3}{2} + 3X_1 \quad \textit{and} \quad f_2(X_1) = \frac{5}{2} - 5X_2.$$

These are plotted as red lines in Figure **??**. Additionally, the black lines in Figure **??** correspond to the estimated partial dependence functions using Algorithm 1.

Clearly, X_2 is more influential than X_1. A natural way of mearing the relative influence of X_2 over X_1 on Y would be to take the absolute value of the ratio of the slopes in $f_2(X_2)$ and $f_1(X_1)$, which in this case gives $5/3 \approx 1.67$. In other words, X_2 is roughly 1.67 times

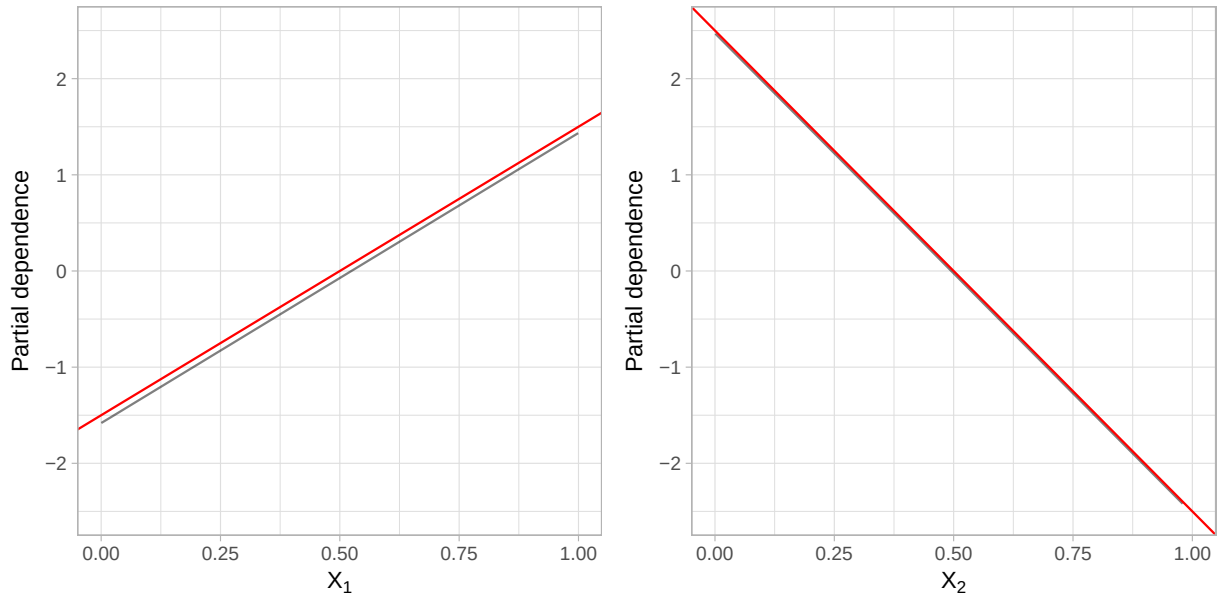

Figure 4: Estimated and true partial dependence plots.

more influential on Y than X_1. Using the partial-dependence-based variable importance metric, we obtain $\text{imp}(X_1) = 1.4828203$ and $\text{imp}(X_2) = 0.8961719$ which gives the ratio $\text{imp}(X_2)/\text{imp}(X_1) \approx 1.65$. In fact, we can compute the true variance for both partial dependence functions:

$$Var[f_1(X_1)] = 3^2/12 \quad and \quad Var[f_2(X_2)] = 5^2/12.$$

Hence, the ratio of the true standard deviations is $5/3 \approx 1.67$.

Using the absolute value of the t-statistic becomes less useful in linear models when, for example, a predictor appears in multiple terms (e.g., interaction effects and polynomial terms). The partial dependence approach, on the other hand, does not suffer from such drawbacks.

3.2 Detecting interaction effects

The same idea can be applied to finding interaction effects. Essentially, a weak interaction effect of x_1 and x_2 on $\mathcal{Y}$ would suggest that $\text{imp}(x_1, x_2)$ has little variation when either x_1 or x_2 is held constant while the other varies.

Let $\boldsymbol{z}_s = (x_i, x_j)$, $i \neq j$, be any two predictors in the feature space $\boldsymbol{x}$. Construct the

partial dependence function $\bar{f}_s(x_i, x_j)$ and compute $\text{imp}(x_i)$ for each unique value of x_j and take the standard deviation of the resulting importance scores. The same can be done for x_j and the results are averaged together. Large values (relative to each other) would be indicative of possible interaction effects.

TODO: How does this relate to Friedman's H-statistic? Do a GBM example and compare the two approaches.

3.3 Computational considerations

Constructing partial dependence functions using the brute force method described in Algorithm 1 can be computationally expensive. Some solutions are discussed in Greenwell (2017). Fortunately, Algorithm 1 is embarrassingly parallel and computing partial dependence functions for each predictor can be done rather quickly on a machine with a multi-core processor. For much larger data sets, it may be worthwhile to reduce the grid size by using specific quantiles for each predictor, rather than using all the unique observations. For example, the partial dependence function can be approximated by using the min and max of each predictor along with the deciles of the unique predictor values.

4 Friedman's regression problem

To further illustrate, we will use one of the regression problems described in Friedman (1991) and Breiman (1996). Inputs are 10 independent variables uniformly distributed on the interval $[0, 1]$; only 5 out of these 10 are actually used. Outputs are created according to the formula

$$\mathcal{Y} = 10 \sin(\pi x_1 x_2) + 20 (x_3 - 0.5)^2 + 10 x_4 + 5 x_5 + \epsilon, \tag{7}$$

where $\epsilon \sim \mathcal{N}(0, \sigma)$. We fit a neural network with one hidden layer containing eight units and a weight decay of 0.01 (these parameters were chosen using 5-fold cross-validation).

Variable importance plots are displayed in Figure **??**. Notice how the Garson and Olden algorithms incorrectly label some of the features not in the true model as "important".

We also constructed the partial dependence functions for all pairwise interactions and computed the interaction statistic discussed in Section 3.2. For each pair (x_i, x_j) we con-

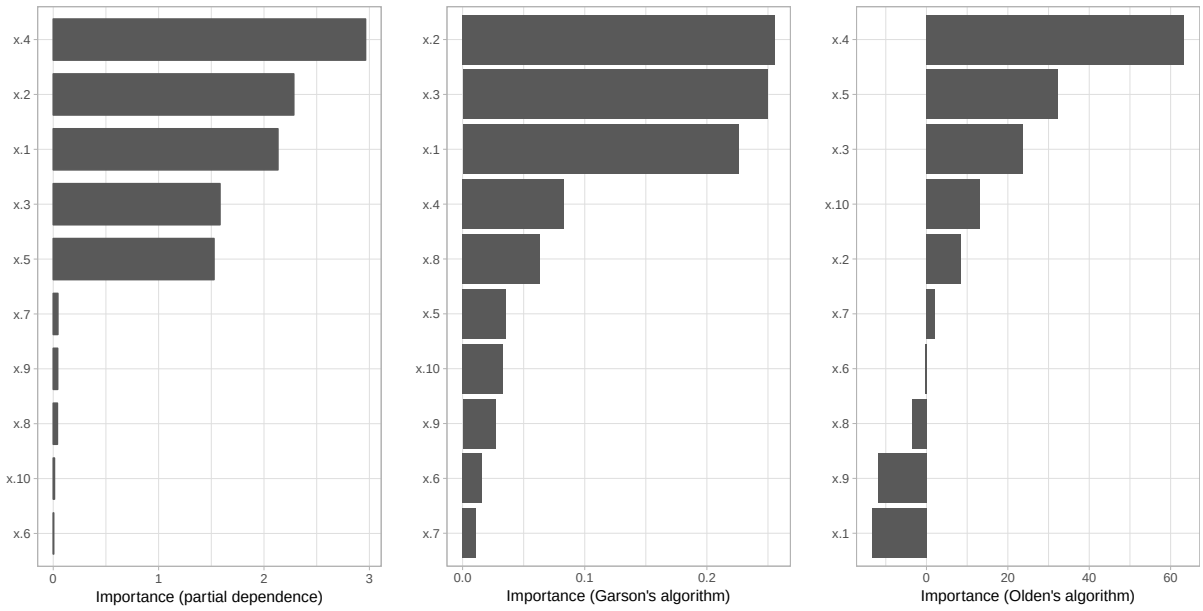

Figure 5: Variable importance plots for the neural network in Figure **??**. *Left*: partial dependence-based algorithm. *Middle*: Garson's algorithm. *Right*: Olden's algorithm.

structed the partial dependence function over a fine grid according to Algorithm 1. Then, we computed $\text{imp}\,(x_i|x_j)$ and $\text{imp}\,(x_j|x_i)$ and averaged the results. The top ten interaction statistics are displayed in Figure **??**. There is a clear indication of an interaction effect between the predictors x_1 and x_2.

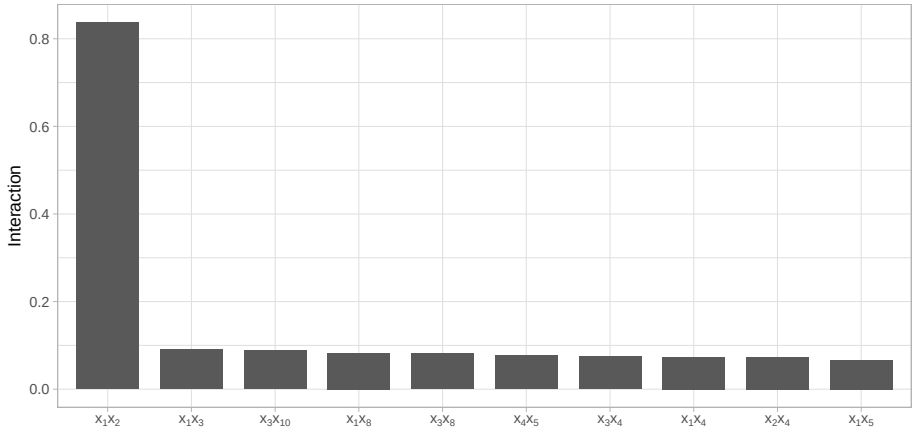

Figure 6: TBD.

4.0.1 Friedman's H-statistic

An alternative measure for the strength of interaction effects is known as Friedman's H-statistic (Friedman and Popescu, 2008). Coincidentally, this statistic is also based on the partial dependence functions of the corresponding predictors.

...

To illustrate, we fit a generalized boosted regression model to the Friedman 1 data set. The parameters were chosen using 5-fold cross-validation. We used the R package `gbm` (Ridgeway, 2017) which has built-in support for computing Friedman's H-statistic for any combination of predictors.

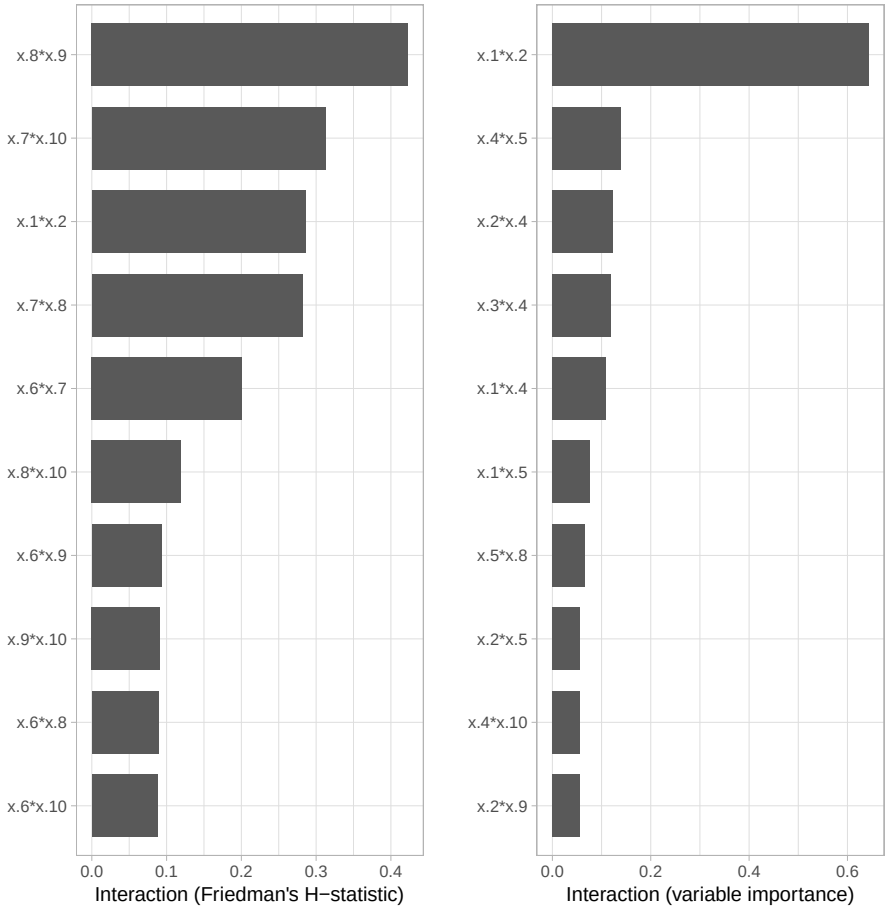

Figure 7: TBD.

5 The Pima indians diabetes data

In this section, we analyze the diabetes test results collected by the the US National Institute of Diabetes and Digestive and Kidney Diseases from a population of women who were at least 21 years old, of Pima Indian heritage, and living near Phoenix, Arizona. The data were taken directly from the `PimaIndiansDiabetes2` data frame which are available in the `mlbench` package (Leisch and Dimitriadou, 2012); for a description of the data and all the variables, see https://archive.ics.uci.edu/ml/datasets/pima+indians+diabetes.

We then used the `nnet` package (**?**) in R to fit a neural network with seven hidden units and a weight decay of 0.01; these were chosen to maximize AUC using repeated 5-fold cross-validation with ten repeats; the cross-validated AUC was 0.77.

The partial dependence functions for all the predictors are displayed in Figure **??**. The PDPs clearly indicate that plasma glucose concentration (`glucose`), diabetes pedigree function (`pedigree`), and 2-hour serum insulin (`insulin`) have the largest effect on the neural network's prediction for the outcome of the diabetes test result (`diabetes`). On the other hand, the PDPs for diastolic blodd pressure (`pressure`) and age (`age`) are relatively flat indicating they contribute less to the predictions than the other predictors. This is further illustrated in the variable importance plot displayed in Figure 9; the importance scores were obtained using Algorithm **??**.

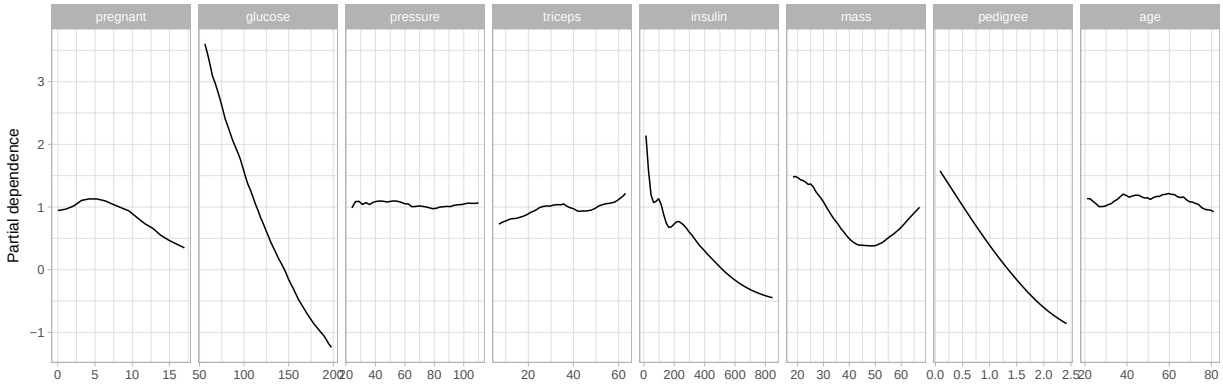

Figure 8: Partial-dependence plots for the eight predictors in the Pima indians data.

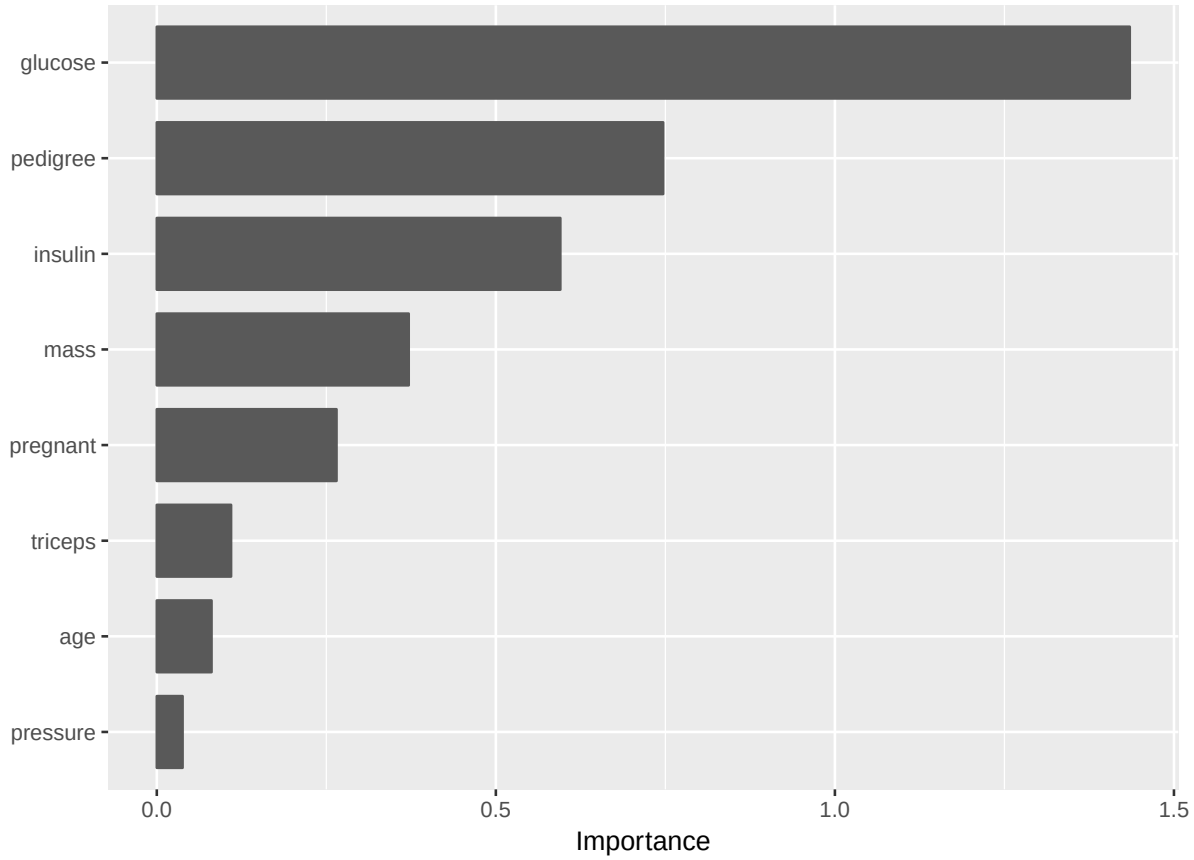

Figure 9: Partial-dependence-based variable importance scores for the eight predictors in the Pima indians data.

6 Conclusion

Coming soon!

SUPPLEMENTARY MATERIAL

R-package `vip`: Contains function for computing variable importance scores and constructing variable importance plots for various types of fitted models in R. The package is hosted on GitHub at https://github.com/AFIT-R/vip.

7 BibTeX

We hope you've chosen to use BibTeX! If you have, please feel free to use the package natbib with any bibliography style you're comfortable with. The .bst file Chicago was used here, and agsm.bst has been included here for your convenience.

References

Breiman, L., J. Friedman, and R. A. O. Charles J. Stone (1984). *Classification and Regression Trees*. The Wadsworth and Brooks-Cole statistics-probability series. Taylor & Francis.

Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association 74*(368), 829–836.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *The Annals of Statistics 19*(1), 1–67.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics 29*, 1189–1232.

Friedman, J. H. and B. E. Popescu (2008). Predictive learning via rule ensembles. *Annals of Applied Statistics 2*(3), 916–954.

Garson, D. G. (1991). Interpreting neural-network connection weights. *Artificial Intelligence Expert 6*(4), 46–51.

Goh, A. (1995). Back-propagation neural networks for modeling complex systems. *Artificial Intelligence in Engineering 9*(3), 143–151.

Greenwell, B. M. (2017). pdp: An r package for constructing partial dependence plots. *The R Journal 9*(1), 421–436.

Harrison, D. and D. L. Rubinfeld (1978). Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management 5*(1), 81–102.

Kuhn, M. and K. Johnson (2013). *Applied Predictive Modeling*. SpringerLink : Bücher. Springer New York.

Leisch, F. and E. Dimitriadou (2012). *mlbench: Machine Learning Benchmark Problems*. R package version 2.1-1.

Liaw, A. and M. Wiener (2002). Classification and regression by randomforest. *R News 2*(3), 18–22.

Olden, J. D., M. K. Joy, and R. G. Death (2004). An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data. *Ecological Modelling 178*(3), 389–397.

Ridgeway, G. (2017). *gbm: Generalized Boosted Regression Models*. R package version 2.1.3.