Understanding Multivariate Adaptive Regression Splines (MARS) with R

Authored by

Mohammed looti

November 6, 2025

Introduction to Multivariate Adaptive Regression Splines (MARS)

The methodology known as Multivariate Adaptive Regression Splines (MARS), initially developed by Jerome H. Friedman, represents a highly effective, non-parametric approach to regression modeling. MARS is expertly designed to identify and model complex, **nonlinear relationships** inherent in data, particularly when the underlying functional form linking the predictor variables to the response variable is completely unknown or highly convoluted. Unlike conventional global modeling techniques, such as standard linear regression, MARS utilizes an adaptive strategy that excels at capturing intricate interactions, structural breaks, and non-monotonic trends, establishing it as an indispensable tool in advanced predictive analytics.

The core innovation of the MARS algorithm lies in its ability to automatically partition the multidimensional data space into smaller, more manageable localized regions. This partitioning is achieved through the systematic identification of optimal split points, referred to as **knots**. Within each localized region, MARS fits a distinct, localized regression model, typically a simple linear function or a constant. This piecewise approach allows the overall model to be highly flexible and adaptive, significantly increasing predictive accuracy for datasets that exhibit heterogeneous structural dynamics or pronounced non-linear behavior across different subsets of the input space.

The practical implementation of the MARS methodology adheres to a structured, two-phase process. First, a large, deliberately overfitted model is constructed in a forward stepwise manner, incorporating numerous basis functions and potential interaction terms to ensure maximum flexibility. Second, a crucial pruning phase employs rigorous statistical criteria, such as generalized cross-validation (GCV) or k-fold cross-validation, to eliminate redundant or weakly contributing terms. This process ensures that the final model achieves an optimal balance between complexity and predictive power, mitigating the risk of **overfitting** while maximizing generalization capability on unseen data.

The fundamental operational sequence for generating a robust MARS model is summarized as follows:

Initiate the forward selection process by iteratively adding pairs of basis functions (hinge functions) corresponding to optimal splitting points (knots) in the predictor variables, aiming for a highly flexible, over-saturated model.

Define the overall model structure using a set of products of these basis functions, effectively capturing both additive effects and potential variable interactions up to a specified degree.

Execute the backward elimination or pruning phase, systematically removing terms that contribute the least to model fit, based on minimization of the generalized cross-validation (GCV) criterion or selection guided by external validation metrics like the Root Mean Squared Error (RMSE).

This comprehensive tutorial will guide you through the practical steps required to implement, optimize, and evaluate a MARS model using the powerful statistical programming language, R, leveraging real-world economic data to illustrate the process.

Setting Up the R Environment and Loading Data

To effectively demonstrate the practical application and optimization of the MARS methodology, we will utilize the publicly accessible **Wage** dataset. This rich, real-world dataset is conveniently packaged within the popular ISLR package, which accompanies the renowned textbook, "Introduction to Statistical Learning with Applications in R." The dataset comprises detailed annual wage information for 3,000 distinct individuals, alongside a comprehensive collection of demographic and employment-related predictor variables, including age, educational attainment level, race, and job classification. Our primary analytical objective is to construct a highly accurate, non-linear model that predicts the quantitative outcome variable, wage, based on these diverse predictors.

Before commencing any statistical modeling endeavor in R, the necessary libraries must be loaded into the active environment. These packages provide the essential infrastructure for efficient data preparation, sophisticated visualization, and, most importantly, the specialized algorithms required for fitting and tuning the MARS model. The suite of packages required includes `dplyr` for robust data wrangling and manipulation, `ggplot2` for high-quality data visualization, and the pivotal `earth` and `caret` packages, which handle the core MARS fitting and rigorous hyperparameter tuning processes, respectively.

Execution of the following R commands ensures that all necessary dependencies are successfully loaded and prepared for the subsequent modeling steps:

library(ISLR) # Contains the foundational Wage dataset
library(dplyr) # Essential for data manipulation and wrangling tasks
library(ggplot2) # Used for generating insightful data visualizations
library(earth) # The specialized package for fitting MARS models
library(caret) # Crucial for tuning model parameters using cross-validation

Exploring the Data Structure

Once the required libraries have been successfully attached to the R session, the next mandatory step involves performing a preliminary inspection of the data structure. This critical exploratory data analysis (EDA) phase is essential for gaining immediate insight into the nature of the variables, including their data types (e.g., factors, numeric integers, or continuous decimals), their scale, and the overall completeness of the observations. A quick view of the initial records allows the analyst

to confirm that the data is loaded correctly and to identify any immediate preparation steps necessary before feeding the variables into the intricate MARS algorithm.

We inspect the first few rows of the `Wage` dataset using the standard R function, which provides a snapshot of the structure and content:

 # View the first six records of the Wage dataset to understand variable structure
head(Wage)

year age maritl race education region
231655 2006 18 1. Never Married 1. White 1. < HS Grad 2. Middle Atlantic
86582 2004 24 1. Never Married 1. White 4. College Grad 2. Middle Atlantic
161300 2003 45 2. Married 1. White 3. Some College 2. Middle Atlantic
155159 2003 43 2. Married 3. Asian 4. College Grad 2. Middle Atlantic
11443 2005 50 4. Divorced 1. White 2. HS Grad 2. Middle Atlantic
376662 2008 54 2. Married 1. White 4. College Grad 2. Middle Atlantic
jobclass health health_ins logwage wage
231655 1. Industrial 1. <=Good 2. No 4.318063 75.04315
86582 2. Information 2. >=Very Good 2. No 4.255273 70.47602
161300 1. Industrial 1. <=Good 1. Yes 4.875061 130.98218
155159 2. Information 2. >=Very Good 1. Yes 5.041393 154.68529
11443 2. Information 1. <=Good 1. Yes 4.318063 75.04315
376662 2. Information 2. >=Very Good 1. Yes 4.845098 127.11574

The resulting output confirms that the dataset is heterogeneous, containing a crucial blend of factor variables (such as `education`, `maritl`, and `jobclass`, which are essentially categorical predictors) and continuous or discrete numeric variables (like `year` and `age`). For our predictive task, we are exclusively focused on the `wage` variable as the target outcome. The mixed data types necessitate that the modeling tool, MARS, is robust enough to handle both types of predictors seamlessly, which the implementation within the earth package is designed to accommodate through dummy variable creation for factors.

Building and Optimizing the MARS Model

The construction of the MARS model is a two-pronged process involving fitting the model structure and simultaneously optimizing its complexity to achieve the best predictive performance. This critical phase relies on rigorous **resampling techniques** to estimate the out-of-sample error accurately. Specifically, we employ **10-fold** k-fold cross-validation (2/5), a highly reliable method, to systematically evaluate various hyperparameter combinations and select the configuration that minimizes the prediction error on held-out folds of the data.

For regression tasks, the standard and most informative metric used for assessing model performance and guiding the hyperparameter search is the Root Mean Squared Error (RMSE) (2/5). The RMSE quantifies the average magnitude of the errors, giving higher weight to large errors, thus providing a comprehensive measure of predictive accuracy. In the context of the MARS implementation provided by the `earth` package, two primary hyperparameters require optimization: `degree`, which dictates the maximum degree of interaction terms allowed (1 for additive, 2 for two-way interactions, and so on), and `nprune`, which controls the final number of basis functions retained after the backward pruning stage.

We define a tuning grid to systematically explore a range of model complexities, ensuring we test models from the simplest additive structure to more complex interactive forms, and from very sparse models to models containing many basis functions. The `caret::train` function then automates the cross-validation and selection process:

```
# Establish a grid of hyperparameters for systematic tuning
hyper_grid <- expand.grid(degree = 1:3,
nprune = seq(2, 50, length.out = 10) %>%
floor())

# Ensure reproducibility of the tuning process through a set seed
set.seed(1)

# Fit the MARS model using 10-fold cross-validation (cv) via the caret package
cv_mars <- train(
x = subset(Wage, select = -c(wage, logwage)),
y = Wage$wage,
method = "earth",
metric = "RMSE",
trControl = trainControl(method = "cv", number = 10),
tuneGrid = hyper_grid)

# Display the specific model configuration that achieved the lowest test RMSE
cv_mars$results %>%
filter(nprune==cv_mars$bestTune$nprune, degree =cv_mars$bestTune$degree)
degree nprune RMSE Rsquared MAE RMSESD RsquaredSD MAESD
1 12 33.8164 0.3431804 22.97108 2.240394 0.03064269 1.4554
```

The detailed analysis of the cross-validation output provides clear guidance for the optimal model structure. The best performance was achieved with a `degree` of 1, strongly suggesting that the relationship between the predictors and wage is primarily additive, relying only on **first-order**

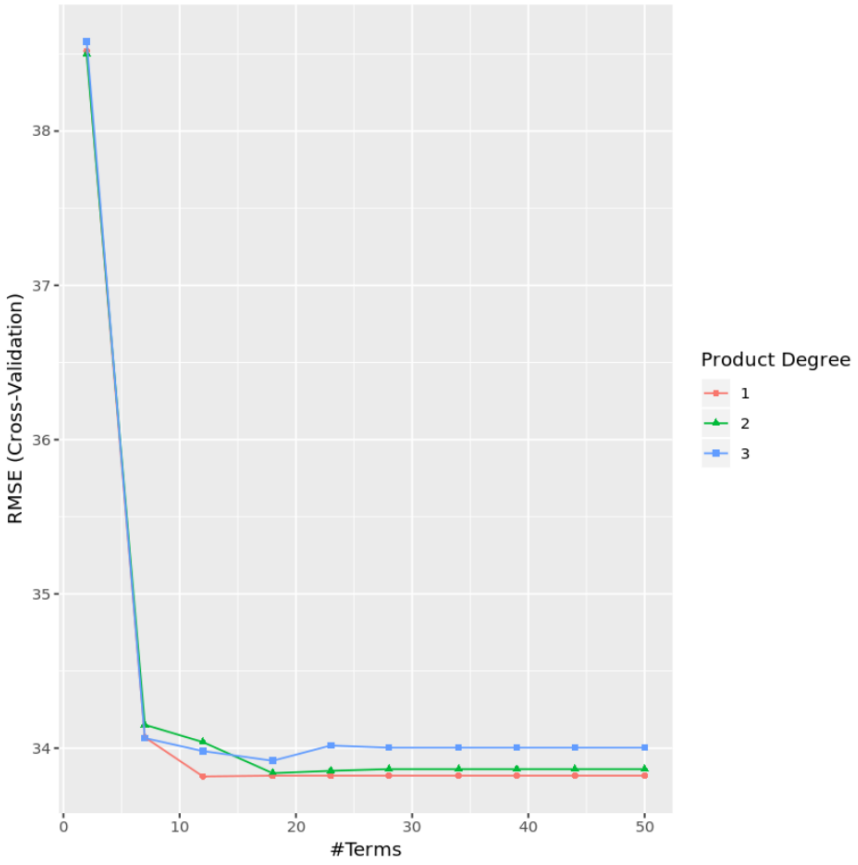
effects, without the need for complex interaction terms between variables to minimize error. Furthermore, an `nprune` value of 12 was selected, indicating that the final pruned model retains 12 essential basis functions. This optimal, parsimonious configuration yielded the lowest predictive error, registering a highly competitive Root Mean Squared Error (RMSE) (3/5) of **33.8164**, confirming a robust and effective fit for the non-linear wage data.

Visualizing Model Performance and Complexity

While numerical summaries like the RMSE are essential, visualizing the relationship between the chosen hyperparameters and the resulting test error provides invaluable, intuitive insight into the model's stability, complexity surface, and adherence to the bias-variance trade-off principle. This visualization serves as a crucial confirmation step, allowing the analyst to graphically verify the selection of the optimal parameters by plotting the Root Mean Squared Error (RMSE) (4/5) against variations in the number of terms (`nprune`) and the maximum degree of interaction (`degree`).

The plotting function built into the `caret` package seamlessly generates this visualization, illustrating how predictive error changes as model complexity increases:

 # Generate a plot showing test RMSE across different tuning parameters
ggplot(cv_mars)



The resulting visualization typically demonstrates a classic pattern: as model complexity (measured by increasing the `degree` or the number of basis functions `nprune`) initially increases, the test error (RMSE) decreases rapidly. However, if the complexity is pushed too far--especially with higher degrees of interaction or an excessive number of terms--the model begins to learn the noise in the training data rather than the underlying signal, a phenomenon known as **overfitting**. This effect is visually evident when the test RMSE curve flattens or begins to rise again. The plot thus serves as a powerful guide, directing the data scientist toward the simplest possible model configuration that still maintains maximum predictive power without compromising its generalization ability.

Contextualizing MARS: Comparison with Other Techniques

While the MARS model (2/5) provides exceptional flexibility and interpretability for modeling complex, nonlinear data structures, its true utility in a professional data science environment is established through rigorous benchmarking. It is a best practice to fit several different types of models to the same dataset and compare their performance based on standardized, out-of-sample metrics, particularly the test error rate determined through robust k-fold cross-validation (3/5). This comparative analysis ensures that the final model selected is not merely adequate, but demonstrably optimal for the prediction task at hand.

MARS is frequently benchmarked against a wide spectrum of regression approaches, spanning foundational statistical models to advanced machine learning paradigms. The fundamental objective remains consistently identifying the model that exhibits the lowest test error, highest stability, and best generalization capability when faced with entirely unseen data. This comparison helps quantify the value added by MARS's non-parametric, adaptive structure relative to more restrictive models.

Models commonly utilized for direct performance comparison against MARS include, but are not limited to:

Multiple Linear Regression: The standard baseline for interpretability and simplicity.

Polynomial Regression: A traditional method for capturing curvature, but less flexible than splines.

Ridge Regression: An L2 regularization technique aimed at stabilizing parameter estimates.

Lasso Regression: An L1 regularization technique useful for automatic feature selection.

Principal Components Regression: A dimension reduction technique used prior to modeling.

Partial Least Squares: A technique focused on finding components that maximize covariance with the response.

By conducting this thorough comparative analysis, data scientists ensure that the final chosen predictive tool is not only mathematically sound but also optimally suited for the predictive task given the specific characteristics and complexity of the dataset. The complete, executable R (2/5) code used to generate the results and visualizations presented in this example is available for review and direct implementation here, encouraging readers to reproduce and extend these findings.