HW8

Team 4

31/10/2019

### Excersise 7.2

Friedman (1991) introduced several benchmark data sets create by simulation. One of these simulations used the following nonlinear equation to create data:

y = 10sin(πx1x2)+20(x3 −0.5)^2 +10x4 +5x5 +N(0,σ^2)

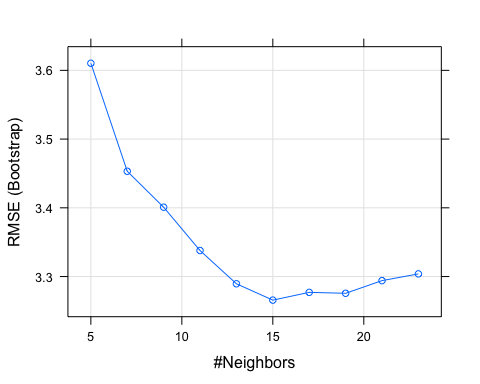
where the x values are random variables uniformly distributed between [0, 1] (there are also 5 other non-informative variables also created in the simulation). The package mlbench contains a function called mlbench.friedman1 that simulates these data:

library(rpart)  
library(dplyr)  
library(tidyr)  
library(caret)  
library(ggplot2)  
library(mlbench)  
set.seed(200)  
trainingData <- mlbench.friedman1(200, sd = 1)  
## We convert the 'x' data from a matrix to a data frame >   
## One reason is that this will give the columns names.   
trainingData$x <- data.frame(trainingData$x)  
## Look at the data using featurePlot(trainingData$x, trainingData$y) or other methods.  
## This creates a list with a vector 'y' and a matrix of predictors 'x'. Also simulate a large test set to > ## estimate the true error rate with good precision:  
testData <- mlbench.friedman1(5000, sd = 1)  
testData$x <- data.frame(testData$x)

Tune several models on these data. Which models appear to give the best performance? Does MARS select the informative predictors (those named X1–X5)?

KNN model

set.seed(921)  
knnModel <- train(x = trainingData$x,  
 y = trainingData$y,  
 method = "knn",  
 preProc = c("center", "scale"),  
 tuneLength = 10)  
plot(knnModel)



knnModel$bestTune

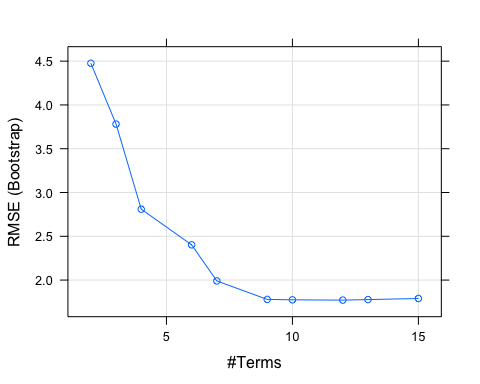
## k  
## 6 15

knnPred <- predict(knnModel, newdata = testData$x)  
postResample(pred = knnPred, obs = testData$y)

## RMSE Rsquared MAE   
## 3.1750657 0.6785946 2.5443169

MARS model

set.seed(921)  
marsModel <- train(x = trainingData$x,  
 y = trainingData$y,  
 method = "earth",  
 tuneLength = 10 )  
plot(marsModel)



marsModel$bestTune

## nprune degree  
## 8 12 1

marsPred <- predict(marsModel, newdata = testData$x)  
postResample(pred = marsPred, obs = testData$y)

## RMSE Rsquared MAE   
## 1.8136467 0.8677298 1.3911836

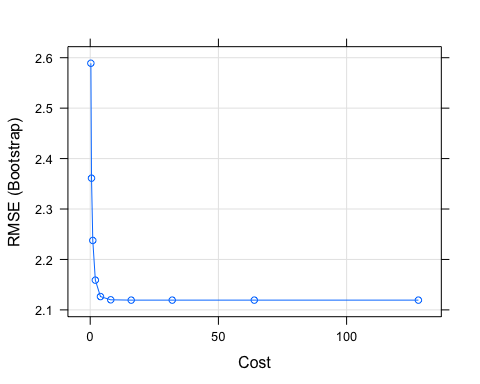
The list of important variables, selected by MARS model

varImp(marsModel)

## earth variable importance  
##   
## Overall  
## X1 100.00  
## X4 84.22  
## X2 67.22  
## X5 45.44  
## X3 34.63  
## X6 11.90  
## X10 0.00  
## X8 0.00  
## X7 0.00  
## X9 0.00

SVM model

set.seed(921)  
svmModel <- train(x = trainingData$x,  
 y = trainingData$y,  
 method = "svmRadial",  
 preProc = c("center", "scale"), tuneLength = 10)  
plot(svmModel)



marsModel$bestTune

## nprune degree  
## 8 12 1

svmModel <- predict(svmModel, newdata = testData$x)  
  
postResample(pred = svmModel, obs = testData$y)

## RMSE Rsquared MAE   
## 2.0856466 0.8237997 1.5847874

MARS model outperformed other models. The worst result has KNN model.

MARS only selected X1 – X6 as important predictors in relationship to the response.

### Excersise 7.5

Exercise 6.3 describes data for a chemical manufacturing process. Use the same data imputation, data splitting, and pre-processing steps as before and train several nonlinear regression models.

1. Which nonlinear regression model gives the optimal resampling and test set performance?
2. Which predictors are most important in the optimal nonlinear regression model? Do either the biological or process variables dominate the list? How do the top ten important predictors compare to the top ten predictors from the optimal linear model?
3. Explore the relationships between the top predictors and the response for the predictors that are unique to the optimal nonlinear regression model. Do these plots reveal intuition about the biological or process predictors and their relationship with yield?

### Approach

The same missing data imputation, data splitting, and pre-processing steps as in exercise 6.3 was used. We are going to train the MARS and SVM nonlinear models and compare results with the optimal PLS linear model. 10 fold cross validation will be used to assess the accuracy of each model on training set.

library(vip)  
library(RANN)  
library(AppliedPredictiveModeling)  
library(dplyr)  
library(tidyr)  
library(e1071)  
library(MASS)  
library(caret)  
library(corrplot)  
library(ggplot2)

# pre-processing the data   
data(ChemicalManufacturingProcess)  
pp <- preProcess(ChemicalManufacturingProcess, method = c("center","scale", "knnImpute", "nzv"))  
ChemicalManufacturingProcess<- predict(pp, ChemicalManufacturingProcess)

# splitting data on training and testing sets  
set.seed(123)  
training.samples <- ChemicalManufacturingProcess$Yield %>%  
createDataPartition(p = 0.8, list = FALSE)  
train.data <- ChemicalManufacturingProcess[training.samples, ]  
test.data <- ChemicalManufacturingProcess[-training.samples, ]

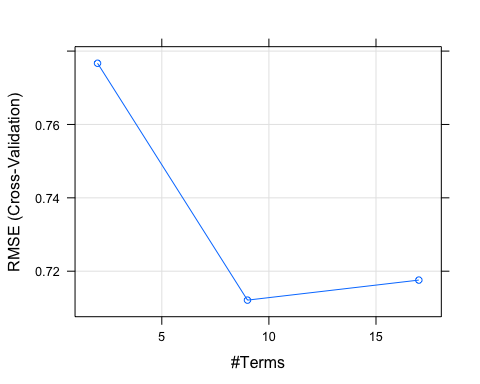
### MARS

Multivariate adaptive regression splines (MARS) provide a convenient approach to capture the nonlinearity relationships in the data by assessing cutpoints (knots) similar to step functions. The procedure assesses each data point for each predictor as a knot and creates a linear regression model with the candidate feature. MARS performs feature importance and selection internally while constructing its model. There are two tuning parameters associated with our MARS model: the degree of interactions and the number of retained terms.

set.seed(123)  
# mars.grid = expand.grid( degree = c( 1:3 ), nprune = c(1:50) )  
ctrl<-trainControl("cv", number = 10)  
mars\_model<- train(Yield~., data = train.data,  
 method = "earth", trControl = ctrl)  
  
# best tuning parameters  
mars\_model$bestTune

## nprune degree  
## 2 9 1

plot(mars\_model)



# testing tuned model on the test set  
predictions <- mars\_model%>% predict(test.data)  
data.frame(  
 RMSE = caret::RMSE(predictions, test.data$Yield),  
 Rsquare = caret::R2(predictions, test.data$Yield)  
)

## RMSE y  
## 1 0.567926 0.6282293

summary(mars\_model)

## Call: earth(x=matrix[144,56], y=c(-1.179,1.226,...), keepxy=TRUE,  
## degree=1, nprune=9)  
##   
## coefficients  
## (Intercept) -0.1799542  
## h(ManufacturingProcess01- -0.717408) 0.2950780  
## h(0.653039-ManufacturingProcess09) -0.5094003  
## h(-1.38676-ManufacturingProcess13) 1.8296310  
## h(-0.827442-ManufacturingProcess32) -1.1172774  
## h(ManufacturingProcess32- -0.827442) 0.6661668  
## h(-1.02435-ManufacturingProcess33) 1.2391404  
## h(0.0324569-ManufacturingProcess39) -0.2545479  
## h(ManufacturingProcess39-0.0324569) -1.4534931  
##   
## Selected 9 of 21 terms, and 6 of 56 predictors  
## Termination condition: RSq changed by less than 0.001 at 21 terms  
## Importance: ManufacturingProcess32, ManufacturingProcess09, ...  
## Number of terms at each degree of interaction: 1 8 (additive model)  
## GCV 0.3418458 RSS 38.2891 GRSq 0.675942 RSq 0.7444016

In general, MARS model performed better on the test set compare to the selected in Exercise 6.3 optimal linear PLS model (RMSE 0.567926 and 1.326151 respectively).

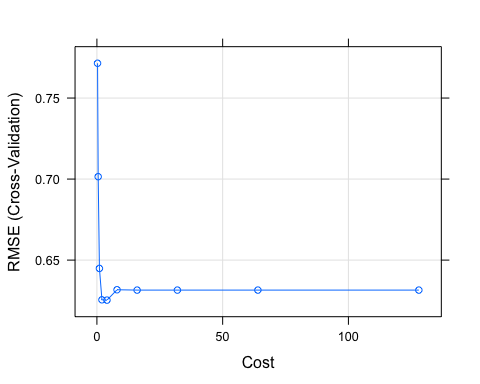
### SVM

SVM is a supervised machine learning algorithm which can be used for classification or regression problems. It uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs. We will consider radial kernel as the radial basis function (as problem requires a non-linear model). The following tuning parameters we will be worked with sigma and C. C is a penalty parameter of the error term. It controls the trade off between smooth decision boundary and classifying the training points correctly.

set.seed(123)  
SVM\_model<- train(Yield~., data = train.data,  
 method = "svmRadial",  
 trControl = ctrl,  
 tuneLength = 10)  
  
# best tuning parameters  
SVM\_model$bestTune

## sigma C  
## 5 0.01347257 4

plot(SVM\_model)



# testing tuned model on the test set  
predictions <- SVM\_model%>% predict(test.data)  
data.frame(  
 RMSE = caret::RMSE(predictions, test.data$Yield),  
 Rsquare = caret::R2(predictions, test.data$Yield)  
)

## RMSE Rsquare  
## 1 0.5964935 0.5483179

SVM model performed better on the test set (lower RMSE) compare to slected in Exercise 6.3 optimal linear PLS model. Also MARS showed slightly better results compare to SVM model (RMSE 0.567926 and 0.5964935 respectively). We can conclude that selected non-linear models such as MARS and SVM significantly outperformed optimal linear PLS model selected in ex.6.3.

Finding the most important variables using selected optimal non-linear model - MARS.

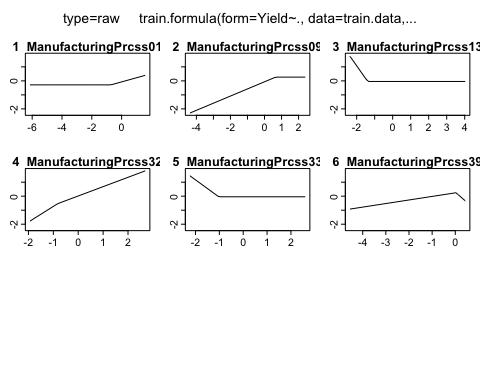
# variable importance plots  
varImp(mars\_model)

## earth variable importance  
##   
## only 20 most important variables shown (out of 56)  
##   
## Overall  
## ManufacturingProcess32 100.00  
## ManufacturingProcess09 65.79  
## ManufacturingProcess13 36.32  
## ManufacturingProcess33 23.92  
## ManufacturingProcess39 19.60  
## ManufacturingProcess01 12.77  
## BiologicalMaterial04 0.00  
## ManufacturingProcess35 0.00  
## ManufacturingProcess34 0.00  
## ManufacturingProcess15 0.00  
## ManufacturingProcess40 0.00  
## ManufacturingProcess30 0.00  
## ManufacturingProcess18 0.00  
## ManufacturingProcess02 0.00  
## ManufacturingProcess06 0.00  
## ManufacturingProcess42 0.00  
## BiologicalMaterial05 0.00  
## BiologicalMaterial11 0.00  
## ManufacturingProcess05 0.00  
## ManufacturingProcess36 0.00

MARS model has selected only 6 variables as important ones and all of them are Manufacturing Process variables, which indicates the dominant importance of Manufacturing Process over Biological Material for the response variable (Yield). Let’s look at the relationships of the most important variables and Yield. Looking at the plots below we can reveal some intuition of Manufacturing processes and Yield, where cutpoints of processes indicate potential for increase/decrease in the yield.

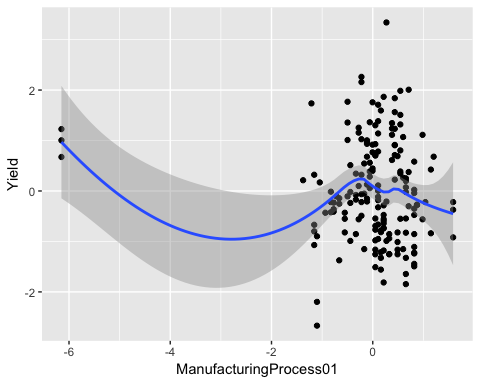
plotmo(mars\_model)

## plotmo grid: BiologicalMaterial01 BiologicalMaterial02  
## -0.07202526 -0.06661163  
## BiologicalMaterial03 BiologicalMaterial04 BiologicalMaterial05  
## -0.08122839 -0.1348209 -0.08059925  
## BiologicalMaterial06 BiologicalMaterial08 BiologicalMaterial09  
## -0.03480919 0.08158206 -0.04830923  
## BiologicalMaterial10 BiologicalMaterial11 BiologicalMaterial12  
## -0.2847816 -0.1811404 -0.006316853  
## ManufacturingProcess01 ManufacturingProcess02 ManufacturingProcess03  
## 0.1056672 0.5096271 0.2872886  
## ManufacturingProcess04 ManufacturingProcess05 ManufacturingProcess06  
## 0.262744 -0.09149682 -0.2599556  
## ManufacturingProcess07 ManufacturingProcess08 ManufacturingProcess09  
## -0.9580199 0.8941637 0.04519175  
## ManufacturingProcess10 ManufacturingProcess11 ManufacturingProcess12  
## -0.1030952 0.02020002 -0.4806937  
## ManufacturingProcess13 ManufacturingProcess14 ManufacturingProcess15  
## 0.09066017 0.1032842 -0.1101043  
## ManufacturingProcess16 ManufacturingProcess17 ManufacturingProcess18  
## 0.0588542 0.04506468 0.07025781  
## ManufacturingProcess19 ManufacturingProcess20 ManufacturingProcess21  
## -0.1360039 0.07604324 -0.1744786  
## ManufacturingProcess22 ManufacturingProcess23 ManufacturingProcess24  
## -0.1218132 -0.01031118 -0.2300722  
## ManufacturingProcess25 ManufacturingProcess26 ManufacturingProcess27  
## 0.07182308 0.06432695 0.06975221  
## ManufacturingProcess28 ManufacturingProcess29 ManufacturingProcess30  
## 0.7255096 -0.066778 0.03954225  
## ManufacturingProcess31 ManufacturingProcess32 ManufacturingProcess33  
## 0.1107323 -0.08632349 0.1836771  
## ManufacturingProcess34 ManufacturingProcess35 ManufacturingProcess36  
## 0.1182687 -0.05513017 -0.5413509  
## ManufacturingProcess37 ManufacturingProcess38 ManufacturingProcess39  
## -0.03063781 0.7174727 0.231727  
## ManufacturingProcess40 ManufacturingProcess41 ManufacturingProcess42  
## -0.4626528 -0.4405878 0.2027957  
## ManufacturingProcess43 ManufacturingProcess44 ManufacturingProcess45  
## -0.1289558 0.2946725 0.1522024



The most important variables that is unique to the optimal nonlinear regression model have non-linear relationship, which was captured by non-linear model better and these variable was identified as important one, for example ManufacturingProcess 01:

ggplot(ChemicalManufacturingProcess, aes(ManufacturingProcess01,Yield)) + geom\_point() + geom\_smooth()



We can conclude that Manufacturing processes is the dominant process over Biological in predicting yield. That means controlling manufacturing processes we can effectively increase yield.