Homework Part 2

Data 624 - Predictive Analytics

16 December 2019

Group Members:

Vinicio Haro

Sang Yoon (Andy) Hwang

Julian McEachern

Jeremy O'Brien

Bethany Poulin

Contents

Getting Started

Overview

Include details on our process in creating this document.

Dependencies

```
# Predicitve Modeling
libraries("AppliedPredictiveModeling", "mice", "caret", "tidyverse",
    "impute", "pls", "caTools", "mlbench", "randomForest", "party",
    "gbm", "Cubist", "rpart")
# Formatting Libraries
libraries("default", "knitr", "kableExtra", "gridExtra", "sqldf",
    "tibble")
# Plotting Libraries
libraries("ggplot2", "grid", "ggfortify", "rpart.plot")

# Data Wrangling
library(AppliedPredictiveModeling)
library(mice)
library(caret)
library(tidyverse)
library(pls)
library(caTools)
library(mlbench)
library(stringr)

# Formatting
library(default)
library(knitr)
library(kableExtra)

# Plotting
library(ggplot2)
library(grid)
library(ggfortify)
library(gridExtra)
```

Assignment 1

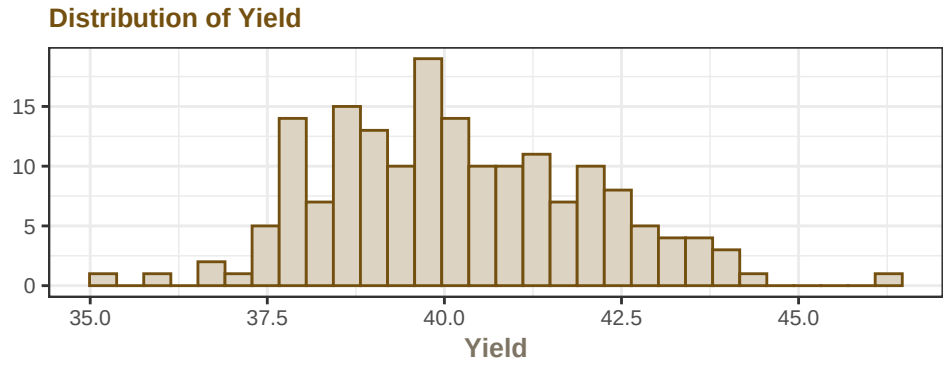
Kuhn and Johnson 6.3

A chemical manufacturing process for a pharmaceutical product was discussed in Sect.1.4. In this problem, the objective is to understand the relationship between biological measurements of the raw materials (predictors), measurements of the manufacturing process (predictors), and the response of product yield. Biological predictors cannot be changed but can be used to assess the quality of the raw material before processing. On the other hand, manufacturing process predictors can be changed in the manufacturing process. Improving product yield by 1% will boost revenue by approximately one hundred thousand dollars per batch:

(a). Start R and use these commands to load the data:

```
data("ChemicalManufacturingProcess")
```

The matrix `processPredictors` contains the 57 predictors (12 describing the input biological material and 45 describing the process predictors) for the 176 manufacturing runs. Yield contains the percent yield for each run. Using a histogram, we examined the distribution of `Yield` and found that the response variable appears unimodal with a normal distribution.



(b). A small percentage of cells in the predictor set contain missing values. Use an imputation function to fill in these missing values (e.g., see Sect. 3.8).

`ManufacturingProcess03` has the largest volume of missing data followed by `ManufacturingProcess11`. Given that each variable has less than 25% of data missing, we should introduce methods of imputation. For our purposes, we will use the MICE method. MICE is formally known as Multiple Imputation with Chained Equations. On a high level, MICE is built off a technique known as the Gibbs sampler.

The Gibbs sampler is a Markov chain based on Monte Carlo. MICE iterates drawing estimates of missing values and parameters related to the distribution of said variables. Chained equations are generally faster than the Monte Carlo based Gibbs sampler. MICE has 5 imputations listed as its default. Predictive mean matching is also a default method for MICE. PMM does a better job at keeping non-linear relationships within individual variables.

In addition to MICE, we drop variables that have near zero variance, however we point out that only one variable was dropped. We still include it as a process step to follow the literature's specifications. After completing MICE, we no longer had missing data in our set. We examined other imputation methods such as KNN but determined that there was no significant change in the summary statistics across different imputation methods.

Table 1: Variables with Missing Values

Predictor	n	Predictor	n
ManufacturingProcess03	15	ManufacturingProcess02	3
ManufacturingProcess11	10	ManufacturingProcess06	2
ManufacturingProcess10	9	ManufacturingProcess01	1
ManufacturingProcess25	5	ManufacturingProcess04	1
ManufacturingProcess26	5	ManufacturingProcess05	1
ManufacturingProcess27	5	ManufacturingProcess07	1
ManufacturingProcess28	5	ManufacturingProcess08	1
ManufacturingProcess29	5	ManufacturingProcess12	1
ManufacturingProcess30	5	ManufacturingProcess14	1
ManufacturingProcess31	5	ManufacturingProcess22	1
ManufacturingProcess33	5	ManufacturingProcess23	1
ManufacturingProcess34	5	ManufacturingProcess24	1
ManufacturingProcess35	5	ManufacturingProcess40	1
ManufacturingProcess36	5	ManufacturingProcess41	1

(c). Split the data into a training and a test set, pre-process the data, and tune a model of your choice from this chapter. What is the optimal value of the performance metric?

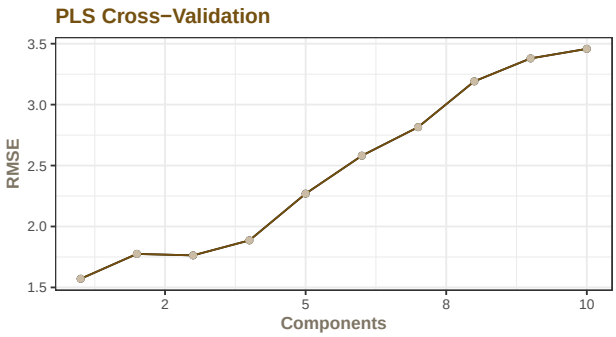
We will build a PLS model also known as partial least squares. PLS is a statistical method that fits a linear regression model by projecting the feature variables and response variable to some new space via a mapping function. Because of this projection mechanism, for both predictors and the response, the method becomes bilinear or simply known as linear with respect to each of the variable types. PLS also has certain advantages over other methods such as being more robust to dealing with issues arising from multicollinearity.

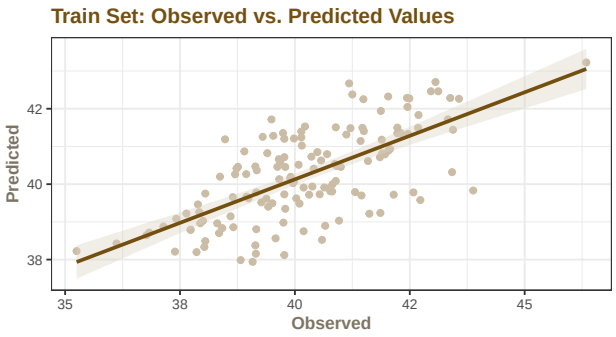
For our PLS model, we partitioned the data by taking 80% of the data as training and the remaining 20% as testing subsets. We also apply center and scaling arguments set to true. We built a standard PLS model and evaluated the root mean summary areas to determine the optimal number of components to select. We generate performance metrics for our best tune below:

Table 2: PLS Performance Metrics on Training Subset

RMSE	Rsquared	MAE
1.571	0.3735	1.1723

Our Baseline PLS model generates a RMSE of 1.57. In addition, the model captures 37.35 % of data variability. We include the visualizations pertaining to the train set cross-validation RMSE tunes and a plot comparing the observed and predicted outcome from our model.





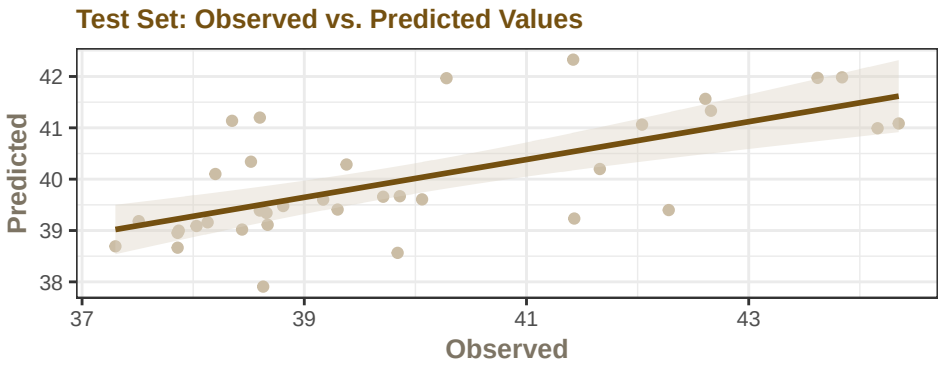
(d). Predict the response for the test set. What is the value of the performance metric and how does this compare with the resampled performance metric on the training set?

We see a decreased R squared against the test data with 44% of the data variability accounted for. We also see the RMSE decrease to 1.55 from our training results of 1.57. There is also a slight increase in the MAE.

Table 3: PLS Performance Metrics on Test Subset

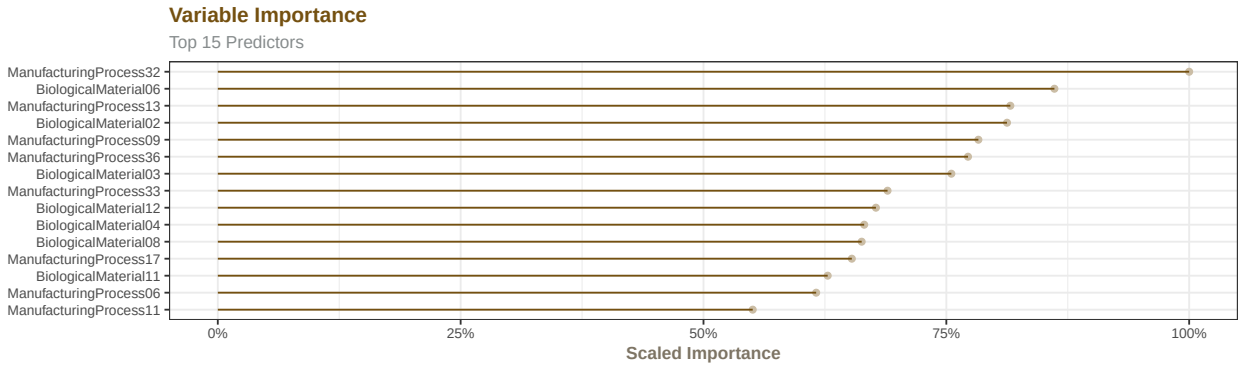
RMSE	Rsquared	MAE
1.5506	0.4426	1.3058

We also plotted the observed and predicted values from our test set against each other below. The deviation from the fitted line tells us that our selected linear model may not provide the best predictions for `Yield`.



(e). Which predictors are most important in the model you have trained? Do either the biological or process predictors dominate the list?

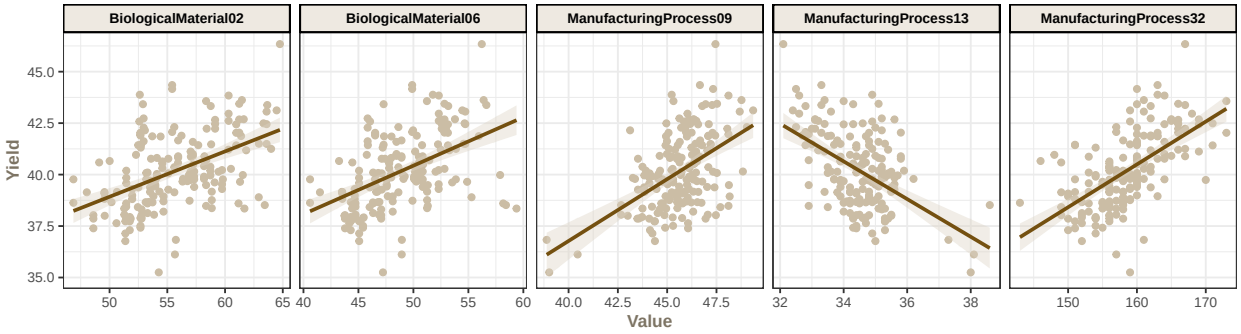
VarImp allows us to identify the variables by name and compute their importance. `ManufacturingProcess32` was flagged as the most important predictor overall and within the group of other Manufacturing Process variables. `BiologicalMaterial06` ranked second and was the most important variable within the BiologicalMaterial group. The variable importance rankings are mixed with 7 variables belonging to Biology and 8 Manufacturing Process variables within the top 15 predictors.



(f). Explore the relationships between each of the top predictors and the response. How could this information be helpful in improving yield in future runs of the manufacturing process?

We used a scatter plot to visualize the relationship between our top five important predictors against our response variable, `Yield`. All but `ManufacturingProcess32` show a moderate positive, linear relationship with yield.

Scatter Plots of Top 5 Predictors Against Yield



We further examined this relationship by analyzing the correlation strength between our top five important response variables with the `Yield`. Out of which, `ManufacturingProcess32` showed the strongest, positive correlation with our response variable.

From a business point of view, our aim is to increase yield since we know that yield ties into revenue. We do not have insight into what mechanics go into each manufacturing process but we can use this knowledge to adjust the processes to emulate the highest yield outputs.

Table 4: Variable Correlation with Yield

Variable	Yield
ManufacturingProcess32	0.6083
ManufacturingProcess09	0.5035
BiologicalMaterial02	0.4815
BiologicalMaterial06	0.4782
ManufacturingProcess13	-0.5037

Assignment 2

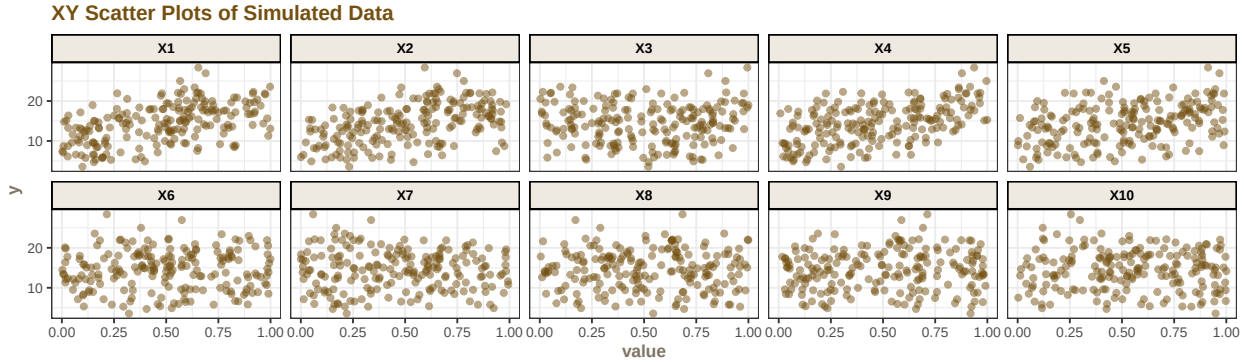
Kuhn and Johnson 7.2

Friedman (1991) introduced several benchmark data sets create by simulation. One of these simulations used the following nonlinear equation to create data: $y = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + N(0,\sigma^2)$; 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. We convert the 'x' data from a matrix to a data frame. One reason is that this will give the columns names. The 'testData' code creates a list with a vector 'y' and a matrix of predictors 'x'. It also simulates a large test set to estimate the true error rate with good precision:

```
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)
testData <- mlbench.friedman1(5000, sd = 1)
testData$x <- data.frame(testData$x)
```



(a). Tune several models on these data. For example:

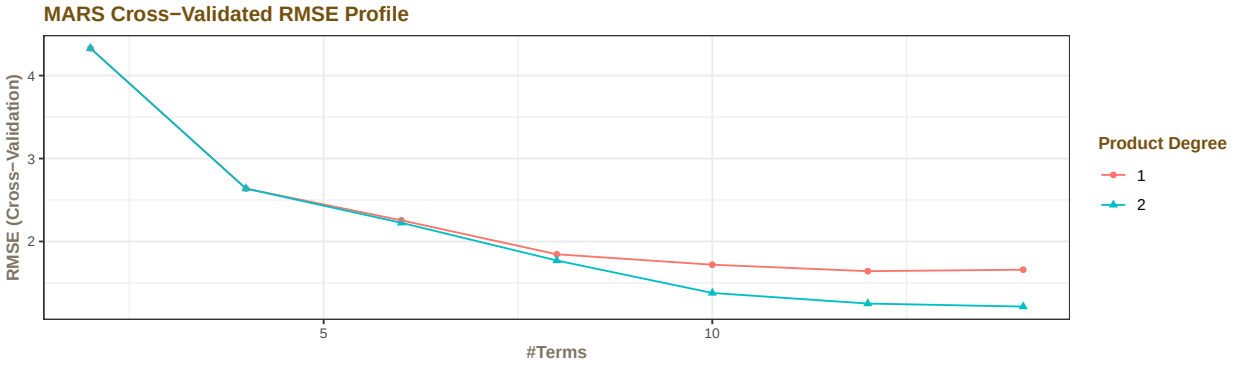
```
knnModel <- train(x = trainingData$x, y = trainingData$y, method = "knn",
    preProc = c("center", "scale"), tuneLength = 10)
knnModel
knnPred <- predict(knnModel, newdata = testData$x)
postResample(pred = knnPred, obs = testData$y)
```

###Model 1-MARS Regression:

MARS, otherwise known as multivariate adaptive regression splines is a non-parametric regression technique that automatically captures non-linearity and interaction between predictors. The basic MARS model has the following form:

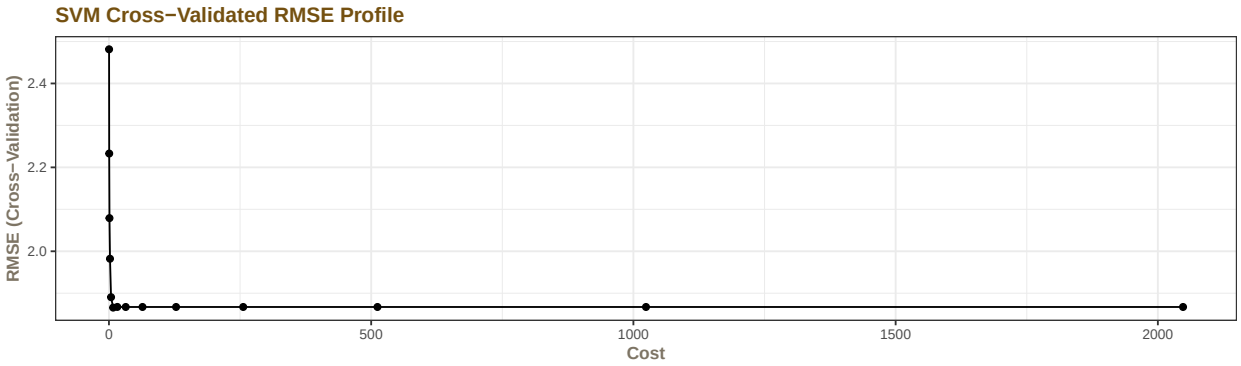
$$\hat{f} = \sum_{i=1}^{k} c_i B_i(x)$$

The model computes the sum of basis functions B multiplied by constant coefficients C.The basis function can either be a constant, a hinge function, or a product of hinge functions. By definition, a hinge function is a piecewise function that converges at a point known as a knot.



###Model 2 SVM:

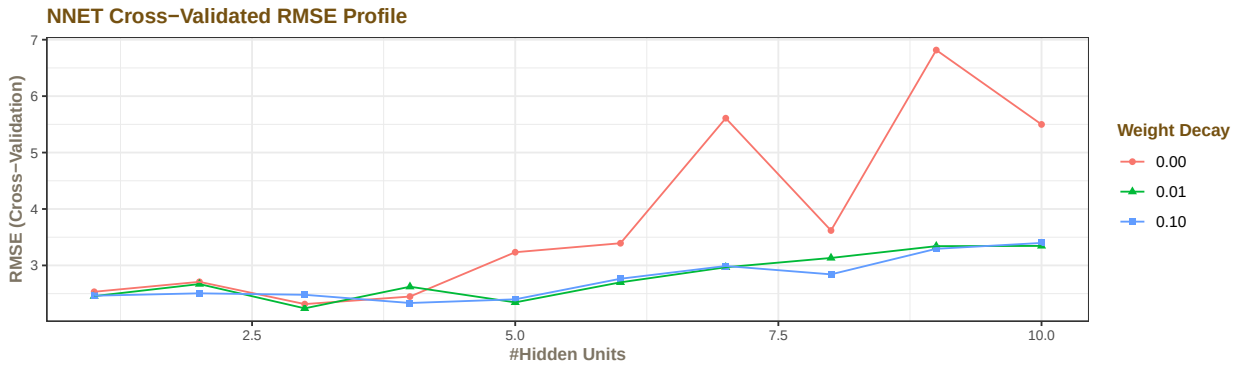
SVM, also known as support vector machine is a method that can be applied to classification and regression tasks. On a high level, SVM creates a hyperplane in n dimensional space. This hyperplane acts like a classification boundary which can be linear or nonlinear. This boundary classifies information from a feature space.



###Model 3 NNET:

NNEt otherwise known as a Neural Network, is a method inspired by a biological neuron system. It uses a system of nodes that are parallel to the way neurons work. It is ideal for capturing non-linear relationships that would otherwise be complicated in most multiple linear regression models. NNET evolves internally based on the calculated weights of each input. The basic structure is shown below:

$$Y = \sum (weight \quad * \quad input) \quad + \quad bias$$

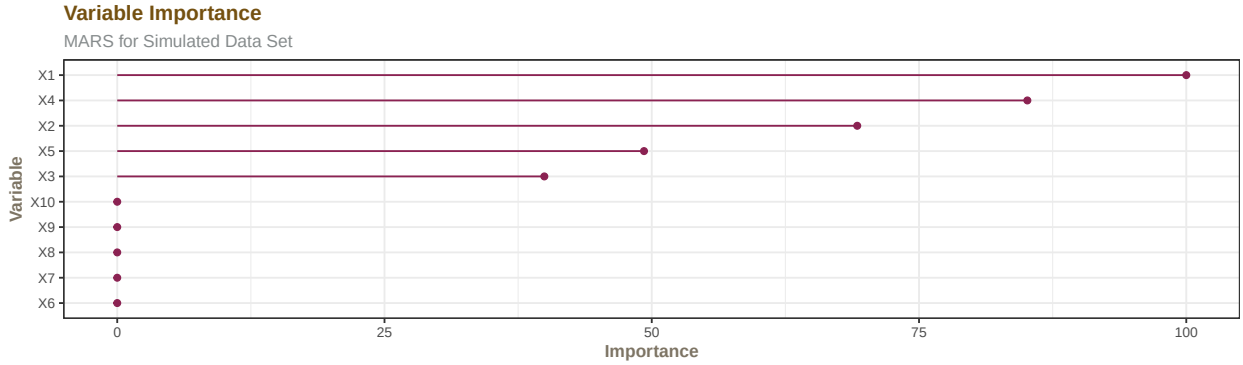


(b). Which models appear to give the best performance? Does MARS select the informative predictors (those named X1-X5)?

Table 5: Model Performance

	RMSE	RSquared	MAE
knnTrain	3.6521	3.6521	3.6521
knnTest	3.2041	0.6820	2.5683
MARSTrain	**4.3290**	**0.9416**	**3.5906**
MARSTest	**1.1723**	**0.9449**	**0.9325**
SVMTrain	2.4814	0.8614	1.9813
SVMTest	2.0425	0.8309	1.5491
NNETTrain	6.8166	0.7991	3.5714
NNETTest	2.9888	0.6526	2.2931

MARS appears to give the best performance based on RMSE, R squared and MAE on test set. The above table shows how our other selected models stack up against the best performing MARS model. We now evaluate the variable importance for our best performing model.



The variable importance table for MARS indicates that variables X1 through X5 were picked as the most important. Out of our collection of important variables used in MARS, X1 is the most important.

It is very likely that the lack of contribution allotted to the X6-X10 variables which bolster the R Squared and RMSE performance and noise from these variables did not reduce the predictive strength of this model as it does in small quantities in the other three models.

Kuhn and Johnson 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.

We pulled in the data processing method from 6.3. This includes imputation and removal of near zero variance features as processing steps. Please refer to 6.3 for a more detailed look at the EDA involved with this data set. We tuned a KNN model, NNET model, MARS, and SVM model using specifications from the literature

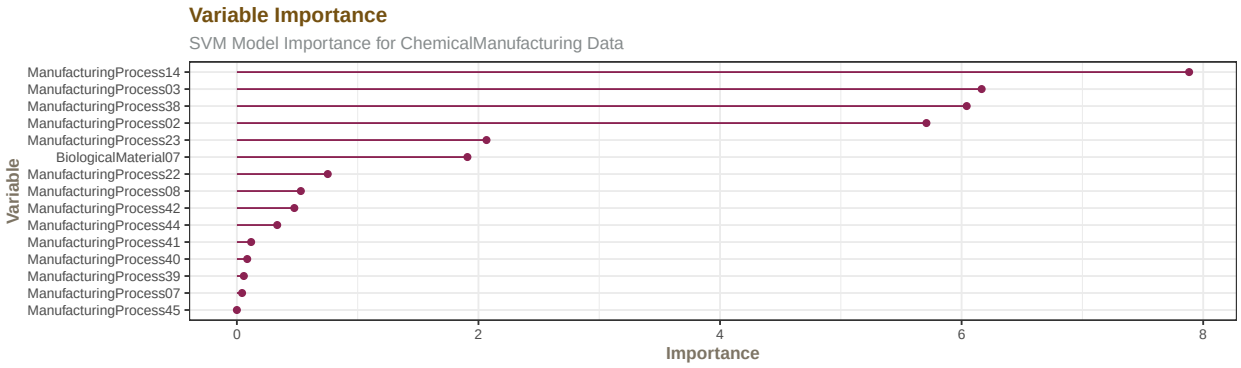
(a). Which nonlinear regression model gives the optimal resampling and test set performance?

Table 6: Model Performance on ChemicalManufacturing Data

	RMSE	RSquared	MAE
knnTrain	1.4554	0.4344	1.1559
knnTest	1.5151	0.4482	1.2684
MARSTrain	1.3809	0.6307	1.0786
MARSTest	1.1999	0.6236	0.9192
SVMTrain	**1.3678**	**0.6546**	**1.1134**
SVMTest	**1.2855**	**0.6030**	**0.9768**
NNETTrain	9.2718	0.3335	5.9237
NNETTest	1.4996	0.4505	1.2194

Radial SVM outperformed the other models across all key KPIs. Radial SVM is generally a more flexible basis function kernel than the base linear kernel. The next best model was MARS regression. In part b, we will address what variables are dominant in our SVM model.

(b). 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?



`ManufacturingProcess` Variables dominate the ranking of important variables with ManufacturingProcess14 at the top. ManufacturingProcess32 was at the top of the important variables list when it came to our linear model with some Biological Process within the top 10.

(c). 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?

Table 7: Correlation

VALUE	Yield
Yield	1.0000
ManufacturingProcess14	-0.0100
ManufacturingProcess38	-0.0865
ManufacturingProcess03	-0.1190
ManufacturingProcess37	-0.1593
ManufacturingProcess02	-0.1954

We examined the top 5 predictors that were flagged as being the most important before the importance measure dropped. There are some pretty clear differences in the data which might explain both the overall poor performance of the linear models as well as the improved significance of Process-Based variables in the non-linear models.

Of the `ManfuacturingProcess` variables, they appear to be either tight clusters or discrete values which predict an array of possible Yields, which is directly opposed the definition of linearly separable data base on earlier examination of correlation plots.

Assignment 3

Kuhn and Johnson 8.1

Recreate the simulated data from Exercise 7.2:

(a). Fit a random forest model to all of the predictors, then estimate the variable importance scores. Did the random forest model significantly use the uninformative predictors (V6-V10)?

The code to the RF model has been provided to us through the literature.

What is the code actually doing? We should dive into the theory. The importance is calculated with the following formula:

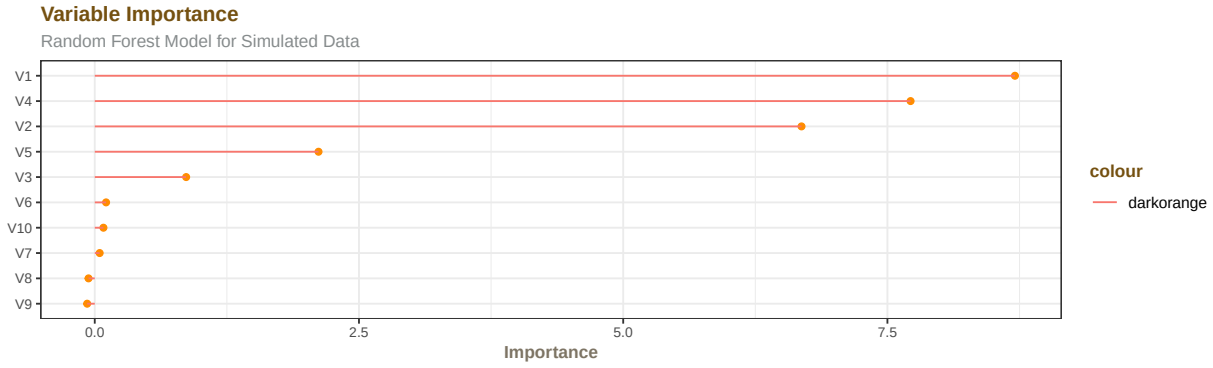
$$ni_j = W_{left(j)}C_{left(j)} - W_{right(j)}C_{right(j)}$$

Let's deconstruct the theory. ni_j stands for node importance. w_j is the weighted number of samples reaching node j. c_j is the impurity value of node j. Left(j) is the child node from left split on node j and right(j) is the child node from right split on node j. Once ni_j is calculated, the importance for each variable on a decision tree is calculated with formula I. Formula I is then normalized as shown as formula II. The final feature importance is the average over all trees. We simply find the sum of the features importance value and then divide by all trees T, shown in formula III.

$$I)\ fi_j = \frac{(\sum_{j\ node\ split\ on\ i})ni_j}{(\sum_{all\ nodes})ni_k}$$

$$II)\ \parallel fi_j \parallel = \frac{(fi_j)}{\sum_{all\ features} fi_j}$$

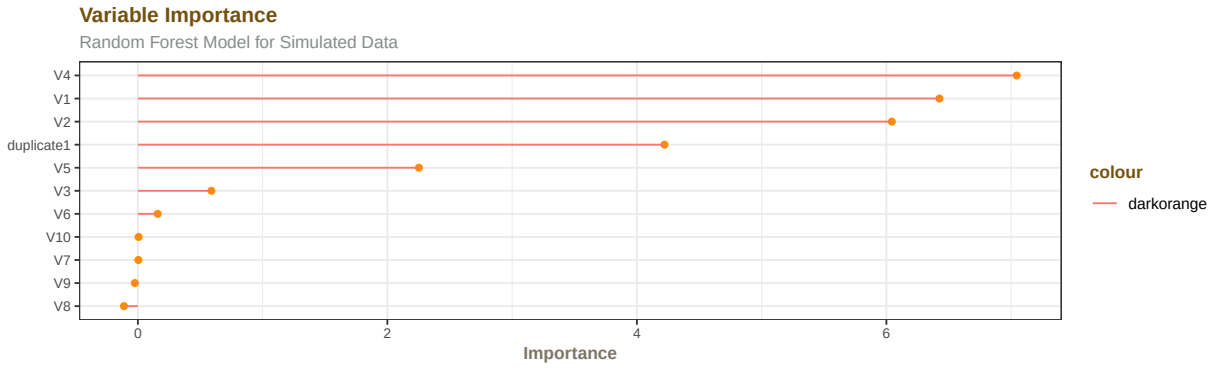
$$III)\ RFfi_j = \frac{(\sum_{all\ trees})\ \parallel fi_j \parallel}{T}$$



Variables V6 to V10 were some of the least important variables, all with a measure less than 1. Variables V1 through V5 were the most important with variable V1 coming in at the top spot.

(b). Now add an additional predictor that is highly correlated with one of the informative predictors. Fit another random forest model to these data. Did the importance score for V1 change? What happens when you add another predictor that is also highly correlated with V1? For example:

We add a feature that has a strong correlation of .93 with existing feature V1. We will call this new feature `duplicate 1`



Note that `V1` decreased importance by roughly 2 measures. V4 is now the most important predictor. It looks like the importance score for `V1` was partly absorbed by new predictor which underestimates true importance of `V1` - the score sum of `V1` and `duplicate1` are similar to the V1 score in (a). It makes sense as `duplicate1` contains almost the same information as `V1`.

(c). Use the 'cforest' function in the party package to fit a random forest model using conditional inference trees. The party package function 'varimp' can calculate predictor importance. The 'conditional' argument of that function toggles between the traditional importance measure and the modified version described in Strobl et al. (2007). Do these importances show the same pattern as the traditional random forest model?

Table 8: Conditional vs Unconditional CForest Model: Variable Importance

features	RF	CF	RF.cor	CF.cor	CF.cond	CF.cor.cond
duplicate1	NA	NA	4.2213	4.5798	NA	0.7636
V1	8.7065	10.0936	6.4252	4.9738	3.3655	0.7965
V2	6.6871	7.5519	6.0433	7.2556	4.6869	4.2718
V3	0.8648	0.0516	0.5894	0.0020	0.0283	0.0062
V4	7.7190	10.3981	7.0444	10.1938	5.8775	5.8944
V5	2.1172	2.2712	2.2531	2.2822	0.7146	0.8100
V6	0.1072	-0.0036	0.1589	-0.0363	0.0099	0.0032
V7	0.0465	0.0576	0.0042	0.0276	0.0335	0.0006
V8	-0.0588	-0.0496	-0.1113	-0.0431	-0.0017	-0.0062
V9	-0.0717	-0.0388	-0.0242	-0.0326	-0.0043	-0.0092
V10	0.0822	0.0045	0.0062	0.0012	0.0112	0.0189

We performed both `varimp(, conditional = T)` and `varimp(, conditional = F)` to compare `varimp` of `cforest` in terms of permutation importance and conditional permutation importance.

1. RF vs CF Given that no correlated term is added, the importance pattern is similar except for the fact that V4 is now the most important feature in CF.
2. RF vs CF (with correlated term added) Given that correlated term is added, the importance score for `duplicate1` is much smaller in CF. This is the pin point difference between importance based on Gini coefficient (decision tree) and permutation test using p-value (conditional inference tree).
3. CF conditional vs CF with correlated term added and conditional When `conditional = T`, we perform conditional permutation test for measuring feature importance instead. Note that `duplicate1` has even smaller importance in `CF.cor.cond` than in `CF.cor`. For `CF.cor.cond`, notice `V1` became 3rd most important feature when it was 2nd

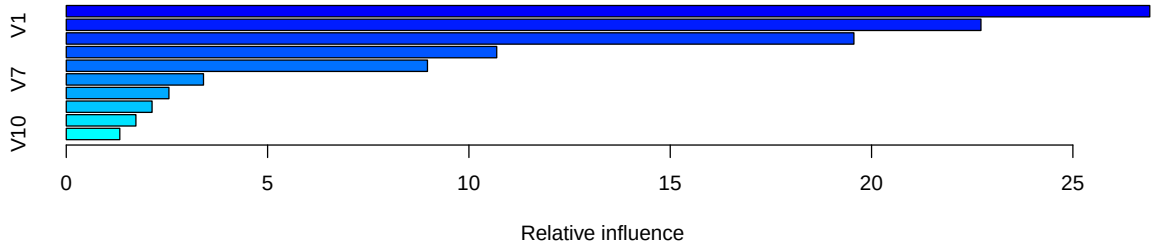
most important for `CF.cor`. This is because conditional permutation helps uncovering the spurious correlation between `V1` and `duplicate1`.

In summary, we learned that `CF` model surpresses the importance score of `duplicate1` which helps maintain the importance of `V1`. When `conditional = TRUE` in `varimp` for `CF` model, the importance score of `duplicate1` is even smaller.

(d). Repeat this process with different tree models, such as boosted trees and Cubist. Does the same pattern occur?

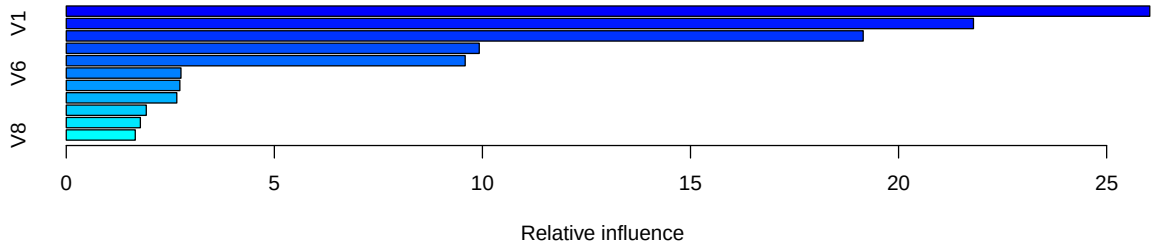
Extracting the relative importance from a BGM object requires the use of the native model summary.

GBM Without Duplicate



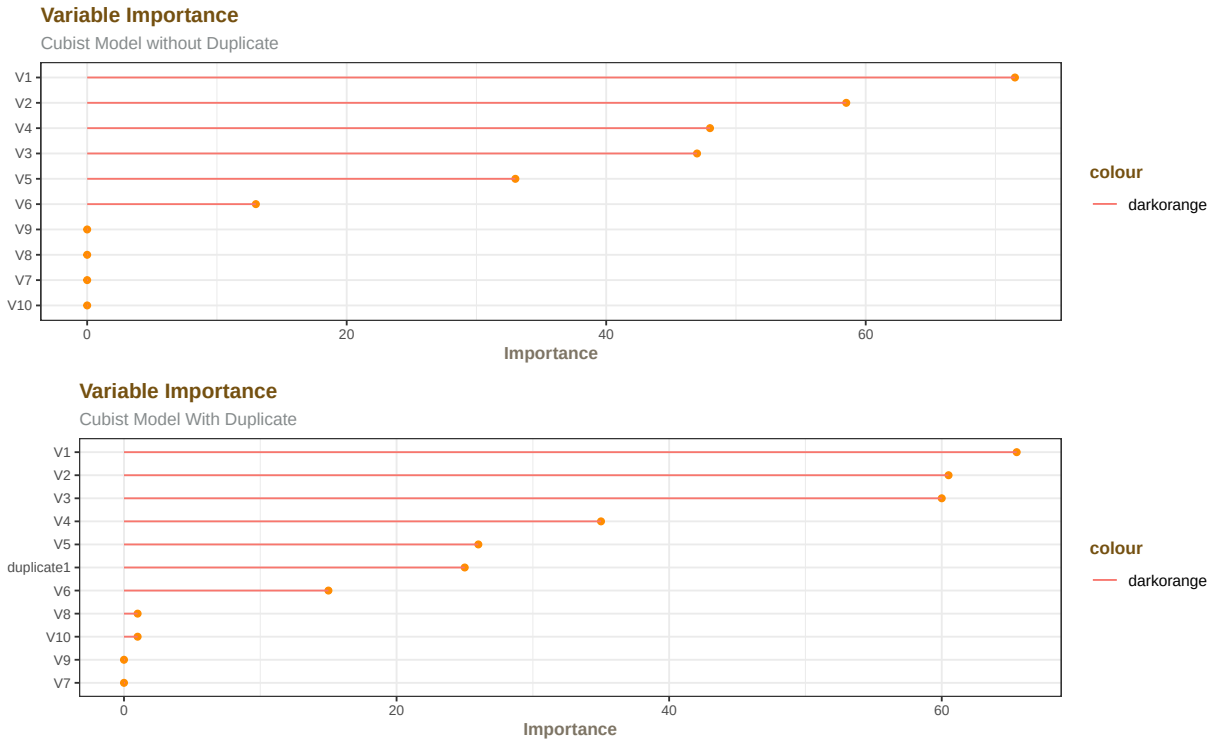

```
    var   rel.inf
V4   V4 26.909998
V1   V1 22.717232
V2   V2 19.562270
V5   V5 10.691070
V3   V3  8.971001
V7   V7  3.408567
V6   V6  2.549782
V8   V8  2.130803
V9   V9  1.729656
V10 V10  1.329621
```

GBM With Duplicate




```
                var   rel.inf
V4               V4 26.036164
```

```
V1                 V1 21.799842
V2                 V2 19.150149
V3                 V3  9.923707
V5                 V5  9.586174
V6                 V6  2.756297
duplicate1 duplicate1  2.729742
V7                 V7  2.656304
V10               V10  1.922812
V9                 V9  1.780691
V8                 V8  1.658119
```

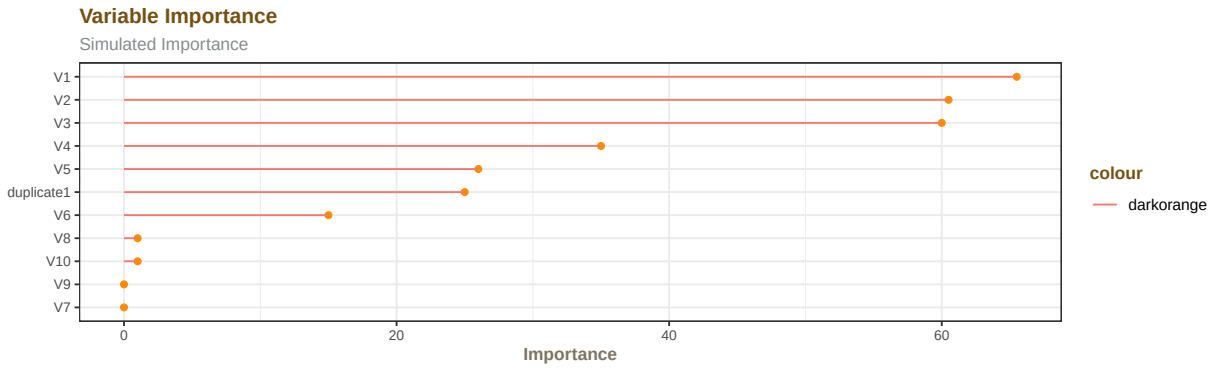


The summary returns the variable name along with a measure of influence on the target variable. From our GBM tree, we can see v4 is the most influential. V1 and correlated duplicate1 are also much more influential. V6-V10 does not break the top half of our list of influential variables.

The next model we want to try is the cubist model. The cubist model is a rather unique variation on trees. Each leaf in the tree contains a linear regression model. Every layer in the tree alters the predicitons used within each leaf contained model. In other words, the selection of predictors in leaf n is based on the previous splis. It should be noted that each intermediate step between leafs also contain a linear model. Predictions are made via linear models on the on the terminal node.

Kuhn and Johnson 8.2

Use a simulation to show tree bias with different granularities.



Basic regression trees split predictor variables into small groups based on response variable. According to the literature, "predictors with a higher number of distinct values are favored over more granular predictors."

Kuhn and Johnson 8.3

In stochastic gradient boosting the bagging fraction and learning rate will govern the construction of the trees as they are guided by the gradient. Although the optimal values of these parameters should be obtained through the tuning process, it is helpful to understand how the magnitudes of these parameters affect magnitudes of variable importance. Figure 8.24 provides the variable importance plots for boosting using two extreme values for the bagging fraction (0.1 and 0.9) and the learning rate (0.1 and 0.9) for the solubility data. The left-hand plot has both parameters set to 0.1, and the right-hand plot has both set to 0.9:

(a). Why does the model on the right focus its importance on just the first few of predictors, whereas the model on the left spreads importance across more predictors?

Because the model on the right has hig learning rate and high bagging fractions, it is doing two things, using 90% of the variables in each tree and using 90% of the error for a given model. Imagine there were ten variables in this model, if you used a bagging fraction of .9, the every tree would have 9 of the ten trees in it, only one variable would be different from the set of 9 trees in the first model each time. The means that most of the possbile break-points in the trees would be the same from tree to tree, only offering new opportunities for initial splits when the most dominating variabl is removed. Because of this, you would very likely find the model making the same first few decisions each time. And with a learning rate of .9, 90% of the error is added in from each tree,this means that in addition to consistently choosing from one or a few initial splits, you are also maximizing their contributions to the models.

Because of these two factors, the trees contributing to the Stockastic Boosted Tree in this example will make very few decisions (maening they will make the same decisions repeately). That lack of variation in initial choices means that the number of paths the learning can take is limited, and with a high learning rate, a core set of variables is selected early on from the trees built very similarly. In essence, the model never has the opportunity to evaluate other possible variables because the greediness of the model makes the same first few choices every time(

(b). Which model do you think would be more predictive of other samples?

The .9, .9 model would likely be overfit to the training data, because the variation in values within those most important variables may not be reflective of the general population. So, this model will be tuned to choose from sample members following the samples distribution of values in those most important features at the expense of recognizing potiential splits in other variables which might be more common in the poolation that the sample.

With a learning rate of .1 and a bagging fraction of .1, the left model is more likely to build truly weak predictors, from smallers sets of variables, consider more distinct breaking points, and therefore extend better to wild data not fully described by the first few variables in the importance summarise_layers

(c). How would increasing interaction depth affect the slope of predictor importance for either model in Fig.8.24?

Increasing the tree depth would affect both models, but differently. The model with bagging fraction and learning rate equal to .1, increasing the number of nodes in the tree would likely increase the importance of the lower variables, creating a less polynomial slope. this is because making more decisions, means giving weight the variables and values where those decisions are made. This would give importance to those variables.

For the model with bagging fraction and learning rate equal to .9, increasing the depth would likely not change the slope of the lower variables much at all just ad new variable or two to the top. The difference is that in this model, with the high fraction and learning rate, we again will still be making most of the same decisions, from tree to tree, so the only increases in importance come from the added nodes, which will be downstream, and they too are highly likely to be the same from tree to tree, such that you will add a importance low on the scale to those variables upon which the new nodes break (or upper variables with a second or third break will grow in importance). So you might see a reshuffling of upper nodes and the slight increase of a one or two less important variables. However, the overall slope, of quickly going to zero will be contrained by the high bagging fraction and learning rate.

Kuhn and Johnson 8.7

Refer to Exercises 6.3 and 7.5 which describe a chemical manufacturing process. Use the same data imputation, data splitting, and pre-processing steps as before and train several tree-based models:

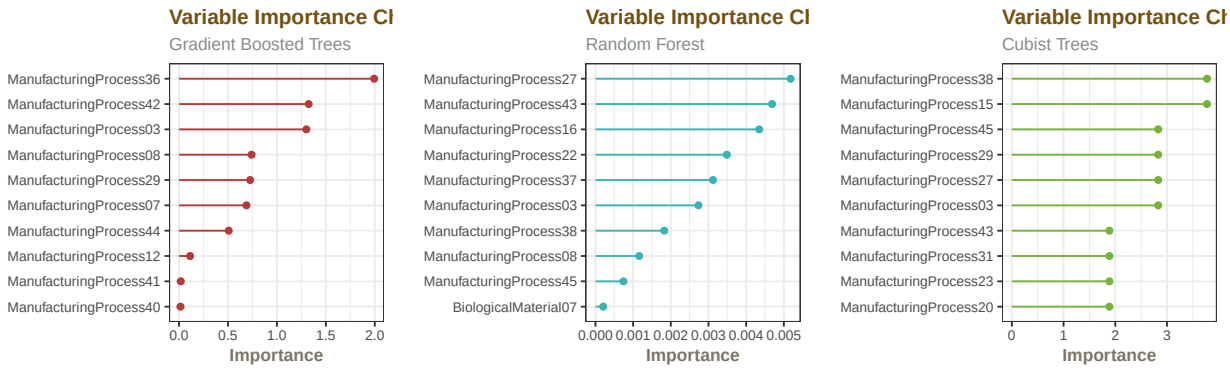
(a). Which tree-based regression model gives the optimal resampling and test set performance?

Table 9: Tree Model Performance on ChemicalManufacturing Data

	RMSE	RSquared	MAE
GBM	1.4978	0.6302	1.1941
GBMTest	1.2073	0.6372	0.9550
RFTrain	1.2292	0.5546	0.9343
RFTest	1.1859	0.6702	0.9286
CubistTrain	**1.0567**	**0.7246**	**0.8118**
CubistTest	**1.0630**	**0.7183**	**0.8187**

The cubist model is the most optimal based on the r squared value. Cubist both on the train and test data has a lower RMSE across the different tree models we selected.

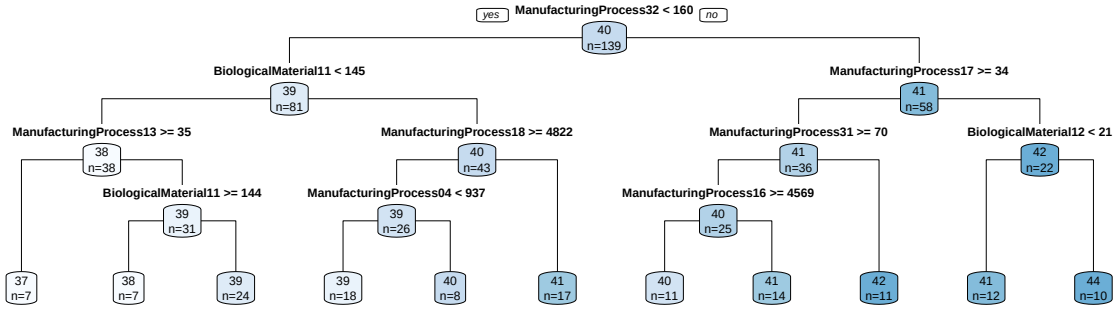
(b). Which predictors are most important in the optimal tree-based regression model? Do either the biological or process variables dominate the list? How do the top 10 important predictors compare to the top 10 predictors from the optimal linear and nonlinear models?



In all the models the Manufacturing processes dominate the list, with slight differences on where in the list and how much influence each has.

Comparing the top 10 variables in each model reveals some strong differences. the Boosted tree follows a rather linear drop-off of importance through a list of exclusively process-based variables. The random forest falls off slower in the first five variables but becomes rapidly linear. All but the last variable in this list are also process based, the last is a biological material variable. The cubist tree, however takes on a very different depreciation, as the values are discrete, with the next six process variables all at exactly thre and the last three variables at 2. Other than the first and last variables all of the cubists top 10 are manufacturing process variable names.

(c). Plot the optimal single tree with the distribution of yield in the terminal nodes. Does this view of the data provide additional knowledge about the biological or process predictors and their relationship with yield?



Based on this regression tree, the differences between process and matrial is that process variables differentiate more observations at each break, which is why the break first, they increase the purity most quickly. However, the final decisions seem to be based rather wholly on biological material variables, such that only looking at Process or prunning too soon, might lead to overfitting.

Assignment 4

TBD

R Script