Data624 - HW2

Joseph Elikishvili

December 9, 2017

## 6.3

#### Chapter 6 talks about Linear regressions. Linear regresions are the most basic but very formidable models. They are easy to implement and can produce very good results. Chapter talks about the ordinary Linear regressions and then dives into PLS and extends conversation to Ridge Regression, Lasso and Elastic nets.

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: 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.

We will get started by loading the necessary libraries, assigning the dataframe to a variable and exploring the dimentions of the dataset

library(AppliedPredictiveModeling)  
library(caret)  
  
data(ChemicalManufacturingProcess)  
df = ChemicalManufacturingProcess  
#summary(df)

### 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).

Next we will impute the missing values. We will evaluate data to see how much data is missing in each of the variables.

colSums(is.na(df))

## Yield BiologicalMaterial01 BiologicalMaterial02   
## 0 0 0   
## BiologicalMaterial03 BiologicalMaterial04 BiologicalMaterial05   
## 0 0 0   
## BiologicalMaterial06 BiologicalMaterial07 BiologicalMaterial08   
## 0 0 0   
## BiologicalMaterial09 BiologicalMaterial10 BiologicalMaterial11   
## 0 0 0   
## BiologicalMaterial12 ManufacturingProcess01 ManufacturingProcess02   
## 0 1 3   
## ManufacturingProcess03 ManufacturingProcess04 ManufacturingProcess05   
## 15 1 1   
## ManufacturingProcess06 ManufacturingProcess07 ManufacturingProcess08   
## 2 1 1   
## ManufacturingProcess09 ManufacturingProcess10 ManufacturingProcess11   
## 0 9 10   
## ManufacturingProcess12 ManufacturingProcess13 ManufacturingProcess14   
## 1 0 1   
## ManufacturingProcess15 ManufacturingProcess16 ManufacturingProcess17   
## 0 0 0   
## ManufacturingProcess18 ManufacturingProcess19 ManufacturingProcess20   
## 0 0 0   
## ManufacturingProcess21 ManufacturingProcess22 ManufacturingProcess23   
## 0 1 1   
## ManufacturingProcess24 ManufacturingProcess25 ManufacturingProcess26   
## 1 5 5   
## ManufacturingProcess27 ManufacturingProcess28 ManufacturingProcess29   
## 5 5 5   
## ManufacturingProcess30 ManufacturingProcess31 ManufacturingProcess32   
## 5 5 0   
## ManufacturingProcess33 ManufacturingProcess34 ManufacturingProcess35   
## 5 5 5   
## ManufacturingProcess36 ManufacturingProcess37 ManufacturingProcess38   
## 5 0 0   
## ManufacturingProcess39 ManufacturingProcess40 ManufacturingProcess41   
## 0 1 1   
## ManufacturingProcess42 ManufacturingProcess43 ManufacturingProcess44   
## 0 0 0   
## ManufacturingProcess45   
## 0

We can see there is a number of missing values, but considering we are dealing with 2570 observations, it is a pretty low number, so imputation will have a very minimal effect.

We will use the missForest to do that, We could use a mean or median or any of the other techniques, but the library does allow us to impute more easily using a range of techniques

library(missForest)  
df\_imp1 = missForest(df)

## missForest iteration 1 in progress...done!  
## missForest iteration 2 in progress...done!  
## missForest iteration 3 in progress...done!

df\_imp = df\_imp1$ximp

Next we will separate out target variable yield from rest of the predictors

data = df\_imp[,2:58]  
target = df\_imp[,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?

Next we will split the data into training and test set. We will use 75/25 ratio to split between the training and test data.

training = createDataPartition( target, p=0.75 )  
predictor\_training = data[training$Resample1,]  
target\_training = target[training$Resample]  
  
predictor\_testing = data[-training$Resample1,]  
target\_testing = target[-training$Resample1]

Next we will use PLS model to train the model, we will center and scale the data using the preProc feauture.

ctrl <- trainControl(method = "cv", number = 10)  
  
plsTune <- train(predictor\_training, target\_training,  
 method = "pls",  
 tuneLength = 20,  
 trControl = ctrl,  
 preProc = c("center", "scale"))

Next We will attempt to find out which of the variables are more important for our model. One way to do it is to evaluate the error metrics at each level, lets take a look

plsTune

## Partial Least Squares   
##   
## 132 samples  
## 57 predictor  
##   
## Pre-processing: centered (57), scaled (57)   
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 120, 117, 119, 120, 119, 118, ...   
## Resampling results across tuning parameters:  
##   
## ncomp RMSE Rsquared MAE   
## 1 1.384646 0.5156225 1.136070  
## 2 1.382239 0.5520846 1.046958  
## 3 1.485286 0.5883230 1.062383  
## 4 1.802570 0.5458733 1.160633  
## 5 2.296458 0.5181616 1.316696  
## 6 2.467796 0.5106514 1.376702  
## 7 2.728478 0.5006842 1.481419  
## 8 2.986280 0.4903481 1.571238  
## 9 3.317428 0.4881181 1.679301  
## 10 3.629848 0.4852466 1.773767  
## 11 3.835952 0.4806501 1.819963  
## 12 3.877151 0.4642502 1.839989  
## 13 3.802287 0.4624730 1.826397  
## 14 3.725779 0.4573299 1.800685  
## 15 3.635713 0.4621838 1.762159  
## 16 3.555753 0.4655657 1.735499  
## 17 3.506166 0.4664287 1.714587  
## 18 3.374584 0.4654890 1.675443  
## 19 3.389976 0.4675826 1.679073  
## 20 3.344234 0.4724176 1.665095  
##   
## RMSE was used to select the optimal model using the smallest value.  
## The final value used for the model was ncomp = 2.

Looks like n=3 yields the best RMSE = 1.405534

### 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?

Next we will use the model to make the predictions and will review the results.

pred\_test = predict(plsTune, predictor\_testing)  
#pred\_train = predict(plsTune, predictor\_training)  
  
pred\_df <- data.frame(obs = target\_testing, pred = pred\_test)  
defaultSummary(pred\_df)

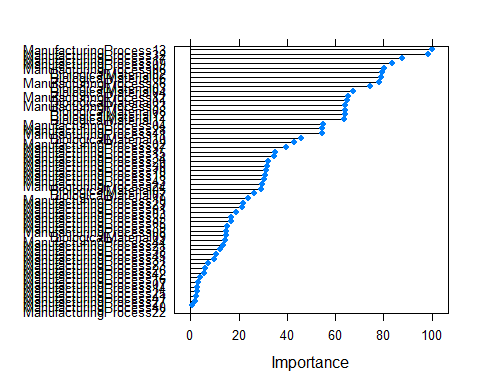
## RMSE Rsquared MAE   
## 3.20626265 0.09498467 1.56550258

We get RMSE = 1.3138455 which is better then the above metric.

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

We can see that RMSE is minimal at n=3 RMSE = 1.460762 Next we can review the importance graphically.

plot(varImp(plsTune))

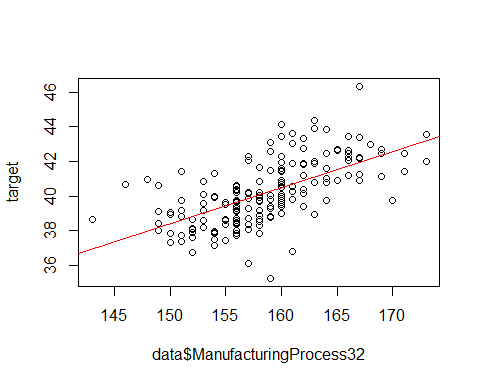


We can see that top predictors are mostly manufacturing processes.The first predictor of bialogical nature is #7, so it is safe to conclude that manufatucring processes have much higher impact on target yield.

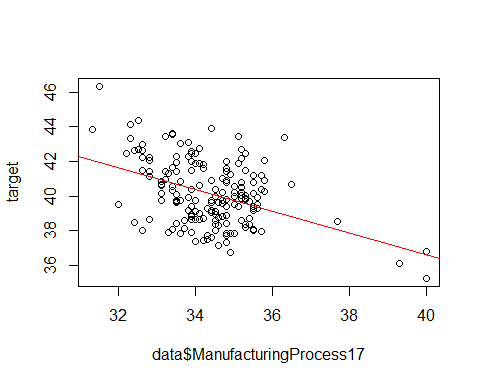
### F. Explore the relationships between each of the top predictors and the response. How could this information be helpful in improving yield in future runs of the manufacturing process?

We will take top 3 predictors and plot them against our target variable.

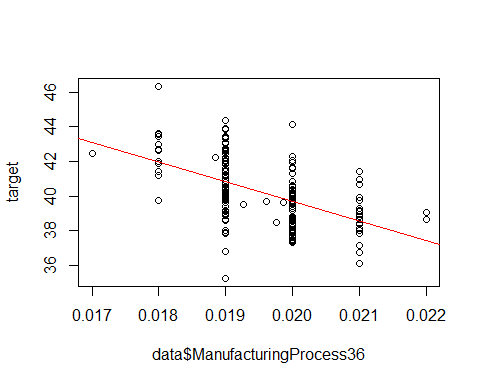
plot(data$ManufacturingProcess32, target)  
abline(lm(target~data$ManufacturingProcess32),col="red",lwd=1.5)



plot(data$ManufacturingProcess17, target)  
abline(lm(target~data$ManufacturingProcess17),col="red",lwd=1.5)



plot(data$ManufacturingProcess36, target)  
abline(lm(target~data$ManufacturingProcess36),col="red",lwd=1.5)



We can see that Manifacturing process32 has a positive corelation with the target and manufacturing process36 and 17 have negative corelation with the target. This means we could adjust the variables to potentially increase the yield.

### Conclusion

Since we have conluded that manufacturing process variables are the most important and have the greatest impact on yield and we have identified the ones we are interested the most and have a corelation information to yield, we belive we can improve the oevral yield.

We would suggest to run more tests with various manifacturing process numbers and to model the most effecient combination which could boost the yield and boost the profits. It is highly likely that such a research would produce positive results and can have a significant impact on sales numbers.