DATA624 Homework 6

Bin Lin

2018-5-3

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:

```
#install.packages("AppliedPredictiveModeling")
library(AppliedPredictiveModeling)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(forecast)
#data(package="AppliedPredictiveModeling")
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.

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).

Approaches: We can use Mice package to impute the missing values. Mice means multivariate imputation by chained equations. To improve the running time, I only run one time of imputation and use the default function ppm. Ppm means predictive mean matching. " It is similar to the regression method except that for each missing value, it imputes a value randomly from a set of observed values whose predicted values are closest to the predicted value for the missing value from the simulated regression model" (Heitjan and Little 1991; Schenker and Taylor 1996).

Interpretation:

The Mice package is very useful in terms of imputating values. As the end result shows, there is no more missing value in the data frame.

```
library(mice)
```

```
## Loading required package: lattice
```

```
a <- mice(ChemicalManufacturingProcess, m = 1, method = "pmm", print = F)
C_M_P <- complete(a)

result<- C_M_P %>%
  select(everything()) %>%
  summarize_all(funs(sum(is.na(.))))

data.frame(sort(result, decreasing = TRUE))
```

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

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?

Approaches: To administer a series of transformations to multiple data sets, the caret class preProcess has the ability to transform, center, scale, or impute values, as well as excluding “near zero-variance” predictors. The function calculates the required quantities for the transformation. After calling the preProcess function, the predict method applies the results to a set of data.

```
#install.packages("e1071")
library(e1071)
library(caret)
```

```
## Loading required package: ggplot2
```

```
CMP_trans <- preProcess(C_M_P, method = c("nzv", "BoxCox", "center", "scale"))
transformed <- predict(CMP_trans, C_M_P)
```

Approaches: The base R function sample can create simple random splits of the data. To create stratified random splits of the data (based on the classes), the createDataPartition function in the caret package can be used. The percent of data that will be allocated to the training set should be specified.

```
set.seed(88)
trainingRows <- createDataPartition(transformed$Yield, p = .80, list= FALSE)

yield_train <- transformed[trainingRows, 1]
predictor_train <- transformed[trainingRows, -1]

yield_test <- transformed[-trainingRows, 1]
predictor_test <- transformed[-trainingRows, -1]
```

Approaches: The tune function of the e1071 package can determine parameter settings using resampling. The train function generates a resampling estimate of performance. Repeated 10-fold cross-validation can be specified with the trainControl function. tuneLength set to the maximum number of dimensions that we want to evaluate.

```
#install.packages("pls")
library(pls)
```

```
##
## Attaching package: 'pls'
```

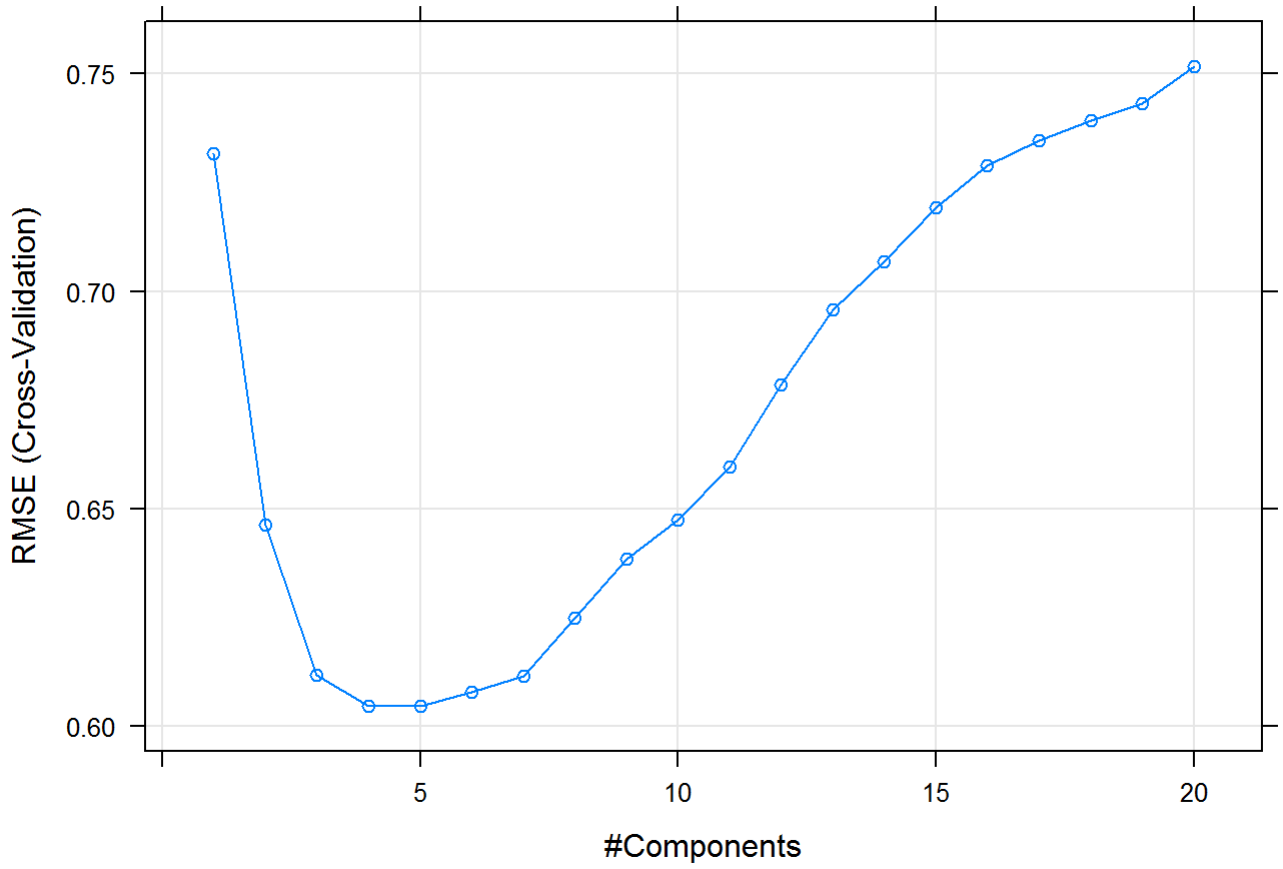
```
## The following object is masked from 'package:caret':
##
##     R2
```

```
## The following object is masked from 'package:stats':
##
##     loadings
```

```
set.seed(888)

ctrl <- trainControl(method = "cv", number = 10)
plsTune <- train(predictor_train, yield_train, method = "pls", tuneLength = 20, trControl = ctr
l)
```

The plot method can be used to visualize the performance profile. RMSE was used to select the optimal model using the smallest value. The optimal model contains 5 components.

```
plot(plsTune)
```



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?

Interpretation: To compute the model product yield values for new samples, the predict method is used. Then use the caret function defaultSummary to estimate the test set performance, The result end up to be 0.6695361. On the other hand, the resampled performance metric on the training set is 0.6080308. They are very closed to each other.

```
CMP_Prediction <- predict(plsTune, predictor_test, ncomp = 1:5)
results <- data.frame(obs = yield_test, pred = CMP_Prediction)
defaultSummary(results)
```

```
##      RMSE  Rsquared       MAE
## 0.6654727 0.5944409 0.5251922
```

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

Approaches: There are two functions that estimate the importance of each predictor in the model: evimp in the earth package and varImp in the caret package. We are going to use varImp.

Interpretation: According to the following list, ManufacturingProcess32 is the most important in the model. Apparently, process predictors dominate the list, because they take up the top 5 positions.

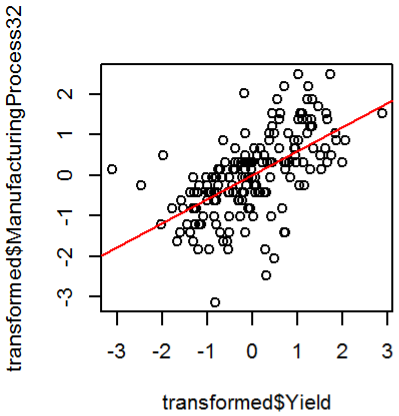
```
varImp(plsTune)
```

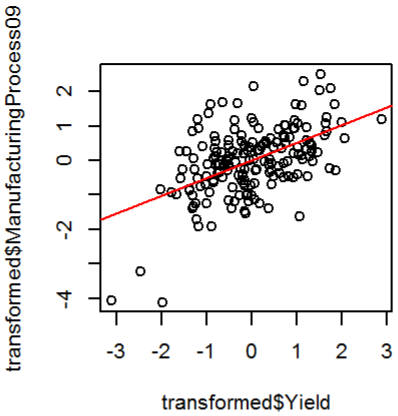
```
## pls variable importance
##
##   only 20 most important variables shown (out of 56)
##
##                         Overall
## ManufacturingProcess32  100.00
## ManufacturingProcess09   87.42
## ManufacturingProcess17   81.64
## ManufacturingProcess13   81.14
## ManufacturingProcess36   75.75
## ManufacturingProcess06   67.47
## BiologicalMaterial02     55.59
## BiologicalMaterial06     55.57
## ManufacturingProcess33   53.09
## ManufacturingProcess34   52.61
## ManufacturingProcess11   51.26
## BiologicalMaterial12     51.09
## BiologicalMaterial03     51.01
## BiologicalMaterial08     48.46
## BiologicalMaterial11     48.44
## BiologicalMaterial04     48.27
## ManufacturingProcess12   45.51
## BiologicalMaterial01     42.44
## ManufacturingProcess37   40.82
## ManufacturingProcess28   37.43
```

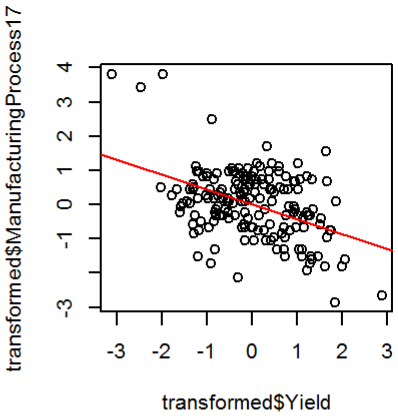
f. Explore the relationships between each of the top predictors and the response. How could this information be helpful in improving yield in future runs of the manufacturing process?

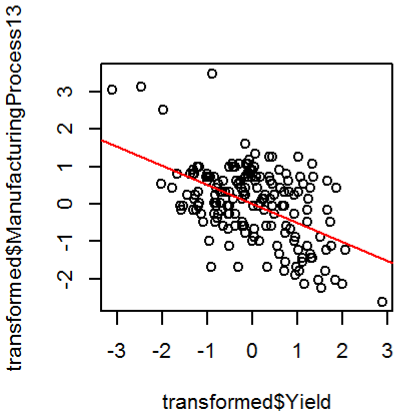
Interpretation: Some of the predictors are positively correlated with the response, while the others are negatively correlated. The information can be helpful because if we improve the predictors with positive correlation, the response will likely get improved and vice versa.

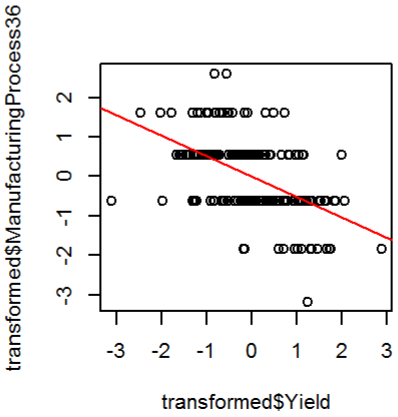
```
par(mfrow = c(2, 3))
plot(transformed$Yield, transformed$ManufacturingProcess32)
abline(lm(transformed$Yield~transformed$ManufacturingProcess32), col="red")

plot(transformed$Yield, transformed$ManufacturingProcess09)
abline(lm(transformed$Yield~transformed$ManufacturingProcess09), col="red")

plot(transformed$Yield, transformed$ManufacturingProcess17)
abline(lm(transformed$Yield~transformed$ManufacturingProcess17), col="red")

plot(transformed$Yield, transformed$ManufacturingProcess13)
abline(lm(transformed$Yield~transformed$ManufacturingProcess13), col="red")

plot(transformed$Yield, transformed$ManufacturingProcess36)
abline(lm(transformed$Yield~transformed$ManufacturingProcess36), col="red")

plot(transformed$Yield, transformed$ManufacturingProcess06)
abline(lm(transformed$Yield~transformed$ManufacturingProcess06), col="red")
```

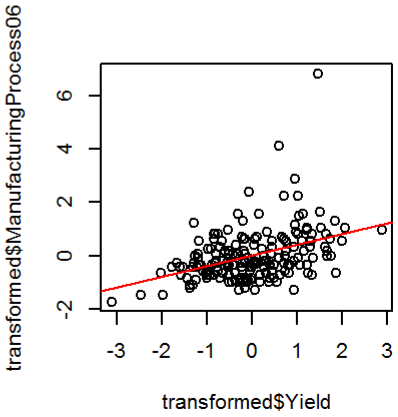

transformed$ManufacturingProcess32
transformed$Yield


transformed$ManufacturingProcess09
transformed$Yield


transformed$ManufacturingProcess17
transformed$Yield


transformed$ManufacturingProcess13
transformed$Yield


transformed$ManufacturingProcess36
transformed$Yield


transformed$ManufacturingProcess06
transformed$Yield