Homework Part Two

Assignment 1: KJ 6.3

Sang Yoon (Andy) Hwang

DATE:2019-10-25

Dependencies

```
# Predicitve Modeling
libraries('AppliedPredictiveModeling', 'caret', 'mice', 'glmnet')

# Formatting Libraries
libraries('default', 'knitr', 'kableExtra')

# Plotting Libraries
libraries('ggplot2', 'grid', 'ggfortify')
```

(1) Kuhn & 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:**

The data contains 176 observations with 58 variables.BiologicalMaterial07 might be a zero variance predictor and we will investigate further.

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

```
'data.frame':   176 obs. of  58 variables:
 $ Yield                 : num  38 42.4 42 41.4 42.5 ...
 $ BiologicalMaterial01  : num  6.25 8.01 8.01 8.01 7.47 6.12 7.48 6.94 6.94 6.94 ...
 $ BiologicalMaterial02  : num  49.6 61 61 61 63.3 ...
 $ BiologicalMaterial03  : num  57 67.5 67.5 67.5 72.2 ...
 $ BiologicalMaterial04  : num  12.7 14.6 14.6 14.6 14 ...
 $ BiologicalMaterial05  : num  19.5 19.4 19.4 19.4 17.9 ...
 $ BiologicalMaterial06  : num  43.7 53.1 53.1 53.1 54.7 ...
 $ BiologicalMaterial07  : num  100 100 100 100 100 100 100 100 100 100 ...
 $ BiologicalMaterial08  : num  16.7 19 19 19 18.2 ...
 $ BiologicalMaterial09  : num  11.4 12.6 12.6 12.6 12.8 ...
 $ BiologicalMaterial10  : num  3.46 3.46 3.46 3.46 3.05 3.78 3.04 3.85 3.85 3.85 ...
 $ BiologicalMaterial11  : num  138 154 154 154 148 ...
 $ BiologicalMaterial12  : num  18.8 21.1 21.1 21.1 21.1 ...
 $ ManufacturingProcess01: num  NA 0 0 0 10.7 12 11.5 12 12 12 ...
 $ ManufacturingProcess02: num  NA 0 0 0 0 0 0 0 0 0 ...
 $ ManufacturingProcess03: num  NA NA NA NA NA NA 1.56 1.55 1.56 1.55 ...
 $ ManufacturingProcess04: num  NA 917 912 911 918 924 933 929 928 938 ...
```

```
$ ManufacturingProcess05: num  NA 1032 1004 1015 1028 ...
$ ManufacturingProcess06: num  NA 210 207 213 206 ...
$ ManufacturingProcess07: num  NA 177 178 177 178 178 177 178 177 177 ...
$ ManufacturingProcess08: num  NA 178 178 177 178 178 178 178 177 177 ...
$ ManufacturingProcess09: num  43 46.6 45.1 44.9 45 ...
$ ManufacturingProcess10: num  NA NA NA NA NA NA 11.6 10.2 9.7 10.1 ...
$ ManufacturingProcess11: num  NA NA NA NA NA NA 11.5 11.3 11.1 10.2 ...
$ ManufacturingProcess12: num  NA 0 0 0 0 0 0 0 0 0 ...
$ ManufacturingProcess13: num  35.5 34 34.8 34.8 34.6 34 32.4 33.6 33.9 34.3 ...
$ ManufacturingProcess14: num  4898 4869 4878 4897 4992 ...
$ ManufacturingProcess15: num  6108 6095 6087 6102 6233 ...
$ ManufacturingProcess16: num  4682 4617 4617 4635 4733 ...
$ ManufacturingProcess17: num  35.5 34 34.8 34.8 33.9 33.4 33.8 33.6 33.9 35.3 ...
$ ManufacturingProcess18: num  4865 4867 4877 4872 4886 ...
$ ManufacturingProcess19: num  6049 6097 6078 6073 6102 ...
$ ManufacturingProcess20: num  4665 4621 4621 4611 4659 ...
$ ManufacturingProcess21: num  0 0 0 0 -0.7 -0.6 1.4 0 0 1 ...
$ ManufacturingProcess22: num  NA 3 4 5 8 9 1 2 3 4 ...
$ ManufacturingProcess23: num  NA 0 1 2 4 1 1 2 3 1 ...
$ ManufacturingProcess24: num  NA 3 4 5 18 1 1 2 3 4 ...
$ ManufacturingProcess25: num  4873 4869 4897 4892 4930 ...
$ ManufacturingProcess26: num  6074 6107 6116 6111 6151 ...
$ ManufacturingProcess27: num  4685 4630 4637 4630 4684 ...
$ ManufacturingProcess28: num  10.7 11.2 11.1 11.1 11.3 11.4 11.2 11.1 11.3 11.4 ...
$ ManufacturingProcess29: num  21 21.4 21.3 21.3 21.6 21.7 21.2 21.2 21.5 21.7 ...
$ ManufacturingProcess30: num  9.9 9.9 9.4 9.4 9 10.1 11.2 10.9 10.5 9.8 ...
$ ManufacturingProcess31: num  69.1 68.7 69.3 69.3 69.4 68.2 67.6 67.9 68 68.5 ...
$ ManufacturingProcess32: num  156 169 173 171 171 173 159 161 160 164 ...
$ ManufacturingProcess33: num  66 66 66 68 70 70 65 65 65 66 ...
$ ManufacturingProcess34: num  2.4 2.6 2.6 2.5 2.5 2.5 2.5 2.5 2.5 2.5 ...
$ ManufacturingProcess35: num  486 508 509 496 468 490 475 478 491 488 ...
$ ManufacturingProcess36: num  0.019 0.019 0.018 0.018 0.017 0.018 0.019 0.019 0.019 0.019 ...
$ ManufacturingProcess37: num  0.5 2 0.7 1.2 0.2 0.4 0.8 1 1.2 1.8 ...
$ ManufacturingProcess38: num  3 2 2 2 2 2 2 2 3 3 ...
$ ManufacturingProcess39: num  7.2 7.2 7.2 7.2 7.3 7.2 7.3 7.3 7.4 7.1 ...
$ ManufacturingProcess40: num  NA 0.1 0 0 0 0 0 0 0 0 ...
$ ManufacturingProcess41: num  NA 0.15 0 0 0 0 0 0 0 0 ...
$ ManufacturingProcess42: num  11.6 11.1 12 10.6 11 11.5 11.7 11.4 11.4 11.3 ...
$ ManufacturingProcess43: num  3 0.9 1 1.1 1.1 2.2 0.7 0.8 0.9 0.8 ...
$ ManufacturingProcess44: num  1.8 1.9 1.8 1.8 1.7 1.8 2 2 1.9 1.9 ...
$ ManufacturingProcess45: num  2.4 2.2 2.3 2.1 2.1 2 2.2 2.2 2.1 2.4 ...
```

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).

After reviewing a few methods of multiple imputation, Multiple Imputation Chained Equations (MICE) was selected for its strength in handling imputation for observations with more than one predictor missing. The method applies one of built-in univariate imputation methods by default `defaultMethod = c("pmm", "logreg", "polyreg", "polr")`.

We will also remove zero variance Predictors using `nearZeroVar`. It diagnoses predictors that have one unique

value (i.e. are zero variance predictors) or predictors that are have both of the following characteristics: they have very few unique values relative to the number of samples and the ratio of the frequency of the most common value to the frequency of the second most common value is large. Indeed, `BiologicalMaterial07` was removed during the process.

> **(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 created 4 models (PLS, Elastic net and LM after center and scale and RLM after PCA). After tuning the models, we have following optimal values.

- **PLS**: number of component = 1
- **Elastic net**: alpha = 1, lambda = 0.23
- **LM**: intecerpt = TRUE
- **RLM**: intercept = TRUE, psi = psi.hampel

In fact, Elastic Net produces a regression model that is penalized with both the Ridge and Lasso. As a mixed model, it shrinks coefficients (like in ridge regression) and set some coefficients to zero (as in LASSO). `glmnet` finds the best combination of alpha and lambda to find the best model. If alpha is between 0 and 1, mixed regularization is used as optimal model. If alpha = 0, RIDGE is chosen as the optimal model. If alpha = 1, LASSO regression is chosen as the optimal model having the lowest RMSE on validation set. In our case, alpha = 1 is chosen for model 2.

```
Partial Least Squares

144 samples
 56 predictor

Pre-processing: centered (56), scaled (56)
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 114, 116, 116, 115, 115
Resampling results across tuning parameters:

  ncomp  RMSE      Rsquared   MAE
   1     1.431015  0.3910776  1.149049
   2     2.276303  0.2944739  1.291454
   3     1.807538  0.3764599  1.198815
   4     1.773229  0.4124674  1.205129
   5     1.895570  0.4187426  1.265041
   6     1.963025  0.4175021  1.291695
   7     2.124636  0.3756893  1.358845
   8     2.313575  0.3441257  1.419544
   9     2.525131  0.3207346  1.491126
  10     2.722476  0.3184469  1.538919

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was ncomp = 1.
```

```
glmnet

144 samples
 56 predictor

Pre-processing: centered (56), scaled (56)
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 116, 115, 115, 116, 114
```

```
Resampling results across tuning parameters:

  alpha  lambda       RMSE      Rsquared   MAE
  0.10   0.002321892  3.493425  0.2767049  1.5813499
  0.10   0.023218918  3.507273  0.3053520  1.4898847
  0.10   0.232189177  2.092907  0.3952821  1.1557505
  0.55   0.002321892  3.308961  0.3053926  1.5183753
  0.55   0.023218918  2.953765  0.3212903  1.3249271
  0.55   0.232189177  1.214322  0.5582371  0.9367045
  1.00   0.002321892  3.133340  0.3536163  1.4412375
  1.00   0.023218918  1.995229  0.4481860  1.0975483
  1.00   0.232189177  1.210932  0.5597812  0.9611315

RMSE was used to select the optimal model using the smallest value.
The final values used for the model were alpha = 1 and lambda = 0.2321892.
```

```
Linear Regression

144 samples
 56 predictor

Pre-processing: centered (56), scaled (56)
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 116, 116, 114, 116, 114
Resampling results:

  RMSE      Rsquared   MAE
  4.357144  0.2521132  1.911584

Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
Robust Linear Model

144 samples
 56 predictor

Pre-processing: principal component signal extraction (56), centered
 (56), scaled (56)
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 114, 115, 116, 116, 115
Resampling results across tuning parameters:

  intercept  psi           RMSE       Rsquared   MAE
  FALSE      psi.huber     40.090048  0.4658447  40.046755
  FALSE      psi.hampel    40.090048  0.4658447  40.046755
  FALSE      psi.bisquare  40.090336  0.4660498  40.047182
   TRUE      psi.huber      1.730719  0.4549443   1.175631
   TRUE      psi.hampel     1.666192  0.4663201   1.144725
   TRUE      psi.bisquare   1.748472  0.4521626   1.183981

RMSE was used to select the optimal model using the smallest value.
The final values used for the model were intercept = TRUE and psi
 = psi.hampel.
```

(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?

It looks like the best model (lowest RMSE on test set and highest R^2) changes from time to time whenever we re-run the model given that each iteration in cross-validation process may give us different result.

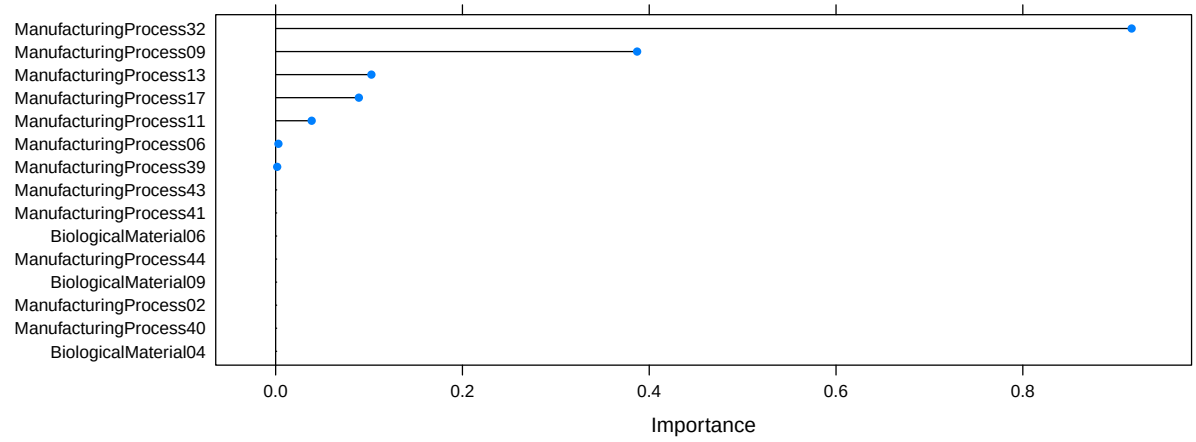
However, when we do resampling 5 times, almost always model2 has the lowest RMSE on test set with highest R^2 in terms of mean which means that model 2 is the best model on average. As we expected, LM model is the worst performer.

We will thus choose model 2 as our final model.

```
        MODEL     RMSE   Rsquare
1         PLS 1.549659 0.4488255
2 Elastic net 1.413493 0.5558069
3          LM 1.448199 0.5167838
4         RLM 1.333060 0.5853321


Call:
summary.resamples(object = resamp)

Models: pls, enet, lm, rlm
Number of resamples: 5

MAE
          Min.   1st Qu.    Median      Mean  3rd Qu.     Max. NA's
pls  0.9650702 1.0021973 1.0642476 1.1490491 1.344048 1.369682    0
enet 0.8963313 0.9014365 0.9134001 0.9611315 1.028219 1.066270    0
lm   1.0791062 1.2787817 1.4011160 1.9115836 2.566259 3.232655    0
rlm  0.8948284 0.9337601 1.0811067 1.1447249 1.118965 1.694964    0

RMSE
         Min.  1st Qu.   Median     Mean  3rd Qu.      Max. NA's
pls  1.220837 1.347966 1.371894 1.431015 1.595709  1.618668    0
enet 1.110943 1.193346 1.203596 1.210932 1.257468  1.289305    0
lm   1.384660 1.551976 1.745738 4.357144 5.374623 11.728724    0
rlm  1.055360 1.149446 1.333219 1.666192 1.453070  3.339865    0

Rsquared
            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
pls  0.145923236 0.2442826 0.4752684 0.3910776 0.5292519 0.5606618    0
enet 0.473582697 0.4777902 0.4803198 0.5597812 0.6088441 0.7583690    0
lm   0.006687546 0.1275771 0.2976102 0.2521132 0.3793288 0.4493624    0
rlm  0.176623096 0.4040727 0.5304717 0.4663201 0.5323915 0.6880417    0
```

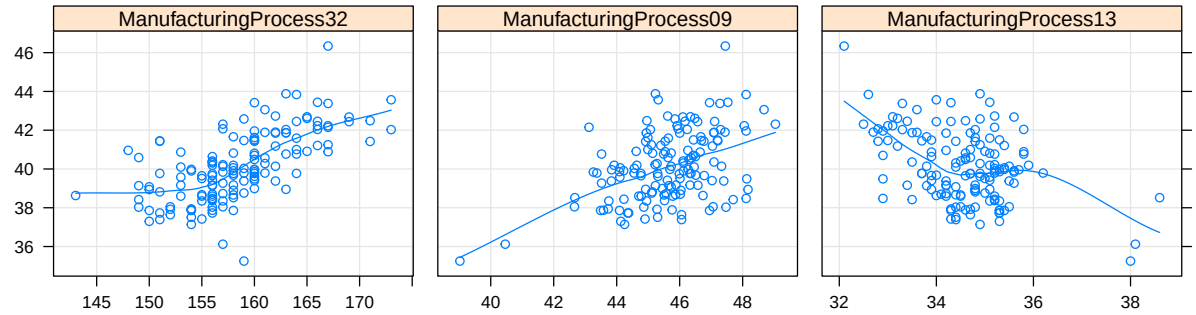
(e). Which predictors are most important in the model you have trained? Do either the biological or process predictors dominate the list?

We see that `ManufacturingProcess32` is the most important in model 2 overall. Among Biological, we know that `BiologicalMaterial06` is the most important which is 4th most important overall.



(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?

From Bivariate plot and correlation matrix, we know that `ManufacturingProcess32` has fairly positive relationship with `Yield` where as other 2 variables have fairly negative relationship. This information can help researchers to focus more on `ManufacturingProcess32` than any other variables if their goal is to increase `Yield`.




```
                         corr_top3
ManufacturingProcess32  0.6490227
ManufacturingProcess09  0.4640182
ManufacturingProcess13 -0.4342814
```

R Code

```r
# (6.3a)
data("ChemicalManufacturingProcess")
str(ChemicalManufacturingProcess)

# (6.3b)
# save df
df <- ChemicalManufacturingProcess

# set seed for split to allow for reproducibility
set.seed(20190227L)
```

```r
# use mice w/ default settings to impute missing data
miceImput <- mice(df, printFlag = FALSE)

# add imputed data to original data set
df_mice <- complete(miceImput)

# Look for any features with no variance:
zero_cols <- nearZeroVar( df_mice )
df_final <- df_mice[,-zero_cols] # drop these zero variance columns

# (6.3c)
# split data train/test
training <- df_final$Yield %>%
  createDataPartition(p = 0.8, list = FALSE)
df_train  <- df_final[training, ]
df_test <- df_final[-training, ]

# model1 - PLS
model1 <- train( Yield~., data = df_train, method="pls",
                 tuneLength=10,
                 preProcess=c("center","scale"), trControl=trainControl(method="cv",number=5) )

# model2 - Elastic net regression
model2 <- train( Yield~., data = df_train, method="glmnet",
                 preProcess=c("center","scale"), trControl=trainControl(method="cv",number=5) )

# model3 - LM
model3 <- train( Yield~., data = df_train, method="lm", preProcess=c("center","scale"), trControl=train

# model4 - RLM with PCA
model4 <- train( Yield~., data = df_train, method="rlm",
                 preProcess=c("pca"), trControl=trainControl(method="cv",number=5) )

# (6.3d)
# Make predictions
p1 <- model1 %>% predict(df_test)
p2 <- model2 %>% predict(df_test)
p3 <- model3 %>% predict(df_test)
p4 <- model4 %>% predict(df_test)

# Model performance metrics
sum_t <- data.frame(
  RMSE1 = caret::RMSE(p1, df_test$Yield),
  RMSE2 = caret::RMSE(p2, df_test$Yield),
  RMSE3 = caret::RMSE(p3, df_test$Yield),
  RMSE4 = caret::RMSE(p4, df_test$Yield),
  Rsquare1 = caret::R2(p1, df_test$Yield),
  Rsquare2 = caret::R2(p2, df_test$Yield),
  Rsquare3 = caret::R2(p3, df_test$Yield),
  Rsquare4 = caret::R2(p4, df_test$Yield)
)
print(sum_t)
```

```r
# resampling 5 times and then get perfomance metrics again
resamp <- resamples( list(pls=model1,enet=model2,lm=model3,rlm=model4) )  # examples of using this are
print( summary(resamp) )

# (6.3e)
varimp <- varImp(model2,scale=F,useModel = T)
plot(varimp, top=15, scales = list(y = list(cex = 0.8)))

# (6.3f)
viporder <- order(abs(varimp$importance),decreasing=TRUE)
topVIP <- rownames(varimp$importance)[viporder[c(1:3)]]

# bivariate relationship
featurePlot(df_train[, topVIP],
            df_train$Yield,
            plot = "scatter",
            between = list(x = 1, y = 1),
            type = c("g", "p", "smooth"),
            layout = c(3,1),
            labels = rep("", 2))

# corr_matrix
corr_top3 <- cor(df_train[, topVIP], df_train$Yield, method = 'pearson', use = 'pairwise.complete.obs')
corr_top3
```