DATA624: Homework 7

Donald Butler

2023-11-05

Contents

Homework 7

Exercise 6.2

Developing a model to predict permeability (see Section 1.4) could save significant resources for a pharmaceutical company, while at the same time more reapidly identifying molecules that have a sufficient permeability to become a drug:

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

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
library(AppliedPredictiveModeling)
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
data(permeability)
```

The matrix `fingerprints` contains the 1,107 binary molecular predictors for the 165 compounds, while `permeability` contains permeability response.

b. The fingerprint predictors indicate the presence or absence of substructures of a molecule and are often sparse meaning that relatively few of the molecules contain each substructure. Filter out the predictors that have low frequencies using the `nearZeroVar` function from the **caret** package. How many predictors are left for modeling?

```r
predictors <- fingerprints[,-nearZeroVar(fingerprints)]

predictors |>
  dim()
```

```
## [1] 165 388
```

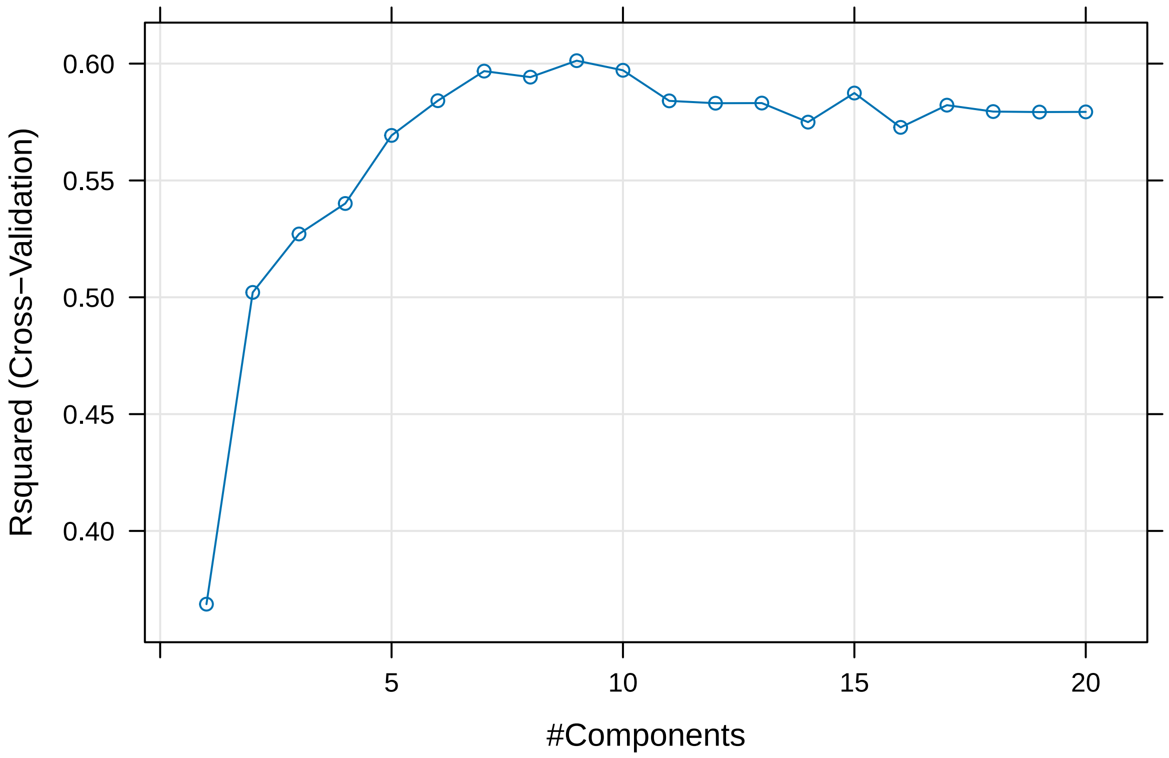
After removing the low frequencies, there are 388 predictors remaining.

c. Split the data into a training and a test set, pre-process the data, and tune a PLS model. How many latent variables are optimal and what is the corresponding re-sampled estimate of R^2?

```r
set.seed(31415)

train <- createDataPartition(permeability, p = .8, list = FALSE)
X_train <- predictors[train,]
X_test <- predictors[-train,]
y_train <- permeability[train]
y_test <- permeability[-train]

plsTune <- train(X_train, y_train, method = 'pls', metric = 'Rsquared',
                 tuneLength = 20, trControl = trainControl(method = 'cv'),
                 preProc = c('center','scale'))

plsTune
```

```
## Partial Least Squares
##
## 133 samples
## 388 predictors
##
## Pre-processing: centered (388), scaled (388)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 121, 119, 119, 121, 118, 120, ...
## Resampling results across tuning parameters:
##
##   ncomp  RMSE      Rsquared   MAE
##    1     13.16678  0.3686642  10.102616
##    2     11.64286  0.5020875   8.027381
##    3     11.80050  0.5271023   8.673277
##    4     11.85865  0.5401392   8.819911
##    5      11.49682  0.5692806   8.495717
##    6      11.14764  0.5841270   8.331981
##    7      10.79245  0.5967975   8.144105
##    8      10.68815  0.5942149   8.223844
##    9      10.51466  0.6012664   8.000850
##   10      10.63739  0.5971470   7.921168
##   11      10.89611  0.5840609   8.168532
##   12      10.91030  0.5830611   8.461339
##   13      11.11168  0.5831191   8.491526
##   14      11.21344  0.5749599   8.491858
##   15      11.03331  0.5874111   8.363449
##   16      11.25919  0.5727178   8.479557
##   17      11.24701  0.5822250   8.420155
##   18      11.27822  0.5794783   8.380912
##   19      11.26692  0.5792757   8.369043
##   20      11.25220  0.5793416   8.210200
##
## Rsquared was used to select the optimal model using the largest value.
## The final value used for the model was ncomp = 9.
```

```r
plot(plsTune)
```



```r
plsTune$results |>
  filter(ncomp == 9)
```

```
##   ncomp     RMSE  Rsquared     MAE   RMSESD RsquaredSD    MAESD
## 1     9 10.51466 0.6012664 8.00085 2.433004  0.1888001 1.831002
```

The optimal tuning had 9 components with $R^2 = 0.6012664$.

d. Predict the response for the test set. What is the test set estimate of R^2?

```
plsPred <- predict(plsTune, newdata = X_test)
postResample(pred = plsPred, obs = y_test)
```

```
##       RMSE   Rsquared        MAE
## 14.0020332  0.1863225 10.1137713
```

$R^2 = 0.1863225$

e. Try building other models discussed in this chapter. Do any have better predictive performance?

PCR

```
pcrTune <- train(X_train, y_train, method = 'pcr', metric = 'Rsquared',
                 tuneLength = 20, trControl = trainControl(method = 'cv'),
                 preProc = c('center','scale'))
pcrPred <- predict(pcrTune, newdata = X_test)
postResample(pred = pcrPred, obs = y_test)
```

```
##      RMSE  Rsquared       MAE
## 13.435755  0.212318  9.198531
```

lars

```
larsTune <- train(X_train, y_train, method = 'lars', metric = 'Rsquared',
                  tuneLength = 20, trControl = trainControl(method = 'cv'),
                  preProc = c('center','scale'))
larsPred <- predict(larsTune, newdata = X_test)
postResample(pred = larsPred, obs = y_test)
```

```
##       RMSE   Rsquared        MAE
## 12.3293046  0.2729282  8.8094694
```

enet

```
enetGrid <- expand.grid(.lambda = c(0, 0.01, .1),
                        .fraction = seq(.05, 1, length = 20))

enetTune <- train(X_train, y_train, method = 'enet', metric = 'Rsquared',
                  tuneGrid = enetGrid, trControl = trainControl(method = 'cv'),
                  preProc = c('center','scale'))
enetPred <- predict(enetTune, newdata = X_test)
postResample(pred = enetPred, obs = y_test)
```

```
##       RMSE   Rsquared        MAE
## 14.6165658  0.1462702 10.3027249
```

The lars model produced the highest R^2 value of 0.273.

f. Would you recommend any of your models to replace the permeability laboratory experiment?

I would recommend the Least Angle Regression (LARS) model since it produced better statistics.

Exercise 6.3

A chemical manufacturing process for a pharmaceutical product was discussed in Section 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.

b. A small percentage of cells in the predictor set contain missing values. Use an imputation function to fill in these missing values.

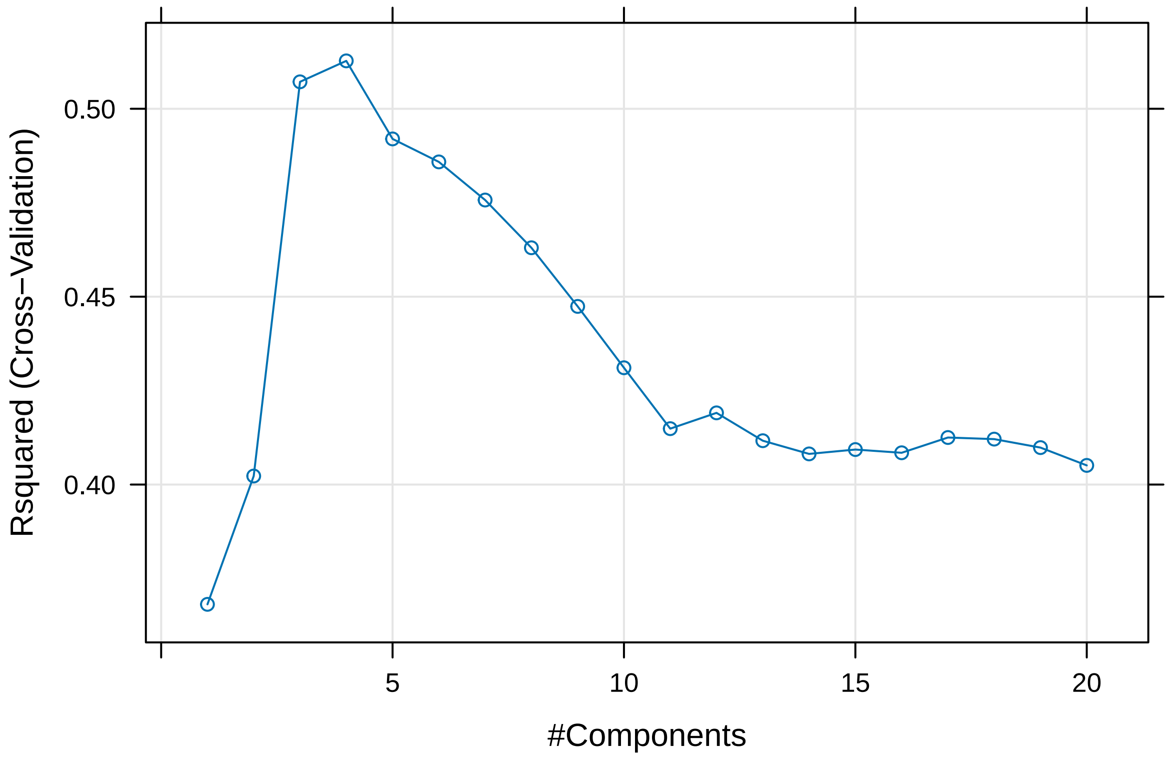
```
imputed <- predict(preProcess(ChemicalManufacturingProcess, method = 'bagImpute'), ChemicalManufacturin
```

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?

```
X <- imputed |>
  select(-Yield)
y <- imputed$Yield

X <- X[,-nearZeroVar(X)]

train <- createDataPartition(y, p = .8, list = FALSE)
X_train <- X[train,]
X_test <- X[-train,]
y_train <- y[train]
y_test <- y[-train]

plsTune <- train(X_train, y_train, method = 'pls', metric = 'Rsquared',
                 tuneLength = 20, trControl = trainControl(method = 'cv'),
                 preProc = c('center','scale'))

plsTune
```

```
## Partial Least Squares
##
## 144 samples
##  56 predictor
##
## Pre-processing: centered (56), scaled (56)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 131, 130, 130, 129, 128, 130, ...
## Resampling results across tuning parameters:
##
##   ncomp  RMSE      Rsquared   MAE
##    1     1.623627  0.3681262  1.247352
##    2     2.024424  0.4022758  1.284387
##    3     1.505117  0.5071746  1.132973
##    4     1.634346  0.5127511  1.158204
##    5     1.911984  0.4919781  1.233227
##    6     2.051749  0.4858665  1.277254
##    7     2.189562  0.4757200  1.328835
##    8     2.267202  0.4630105  1.361028
##    9     2.474943  0.4474028  1.425831
##   10     2.733383  0.4310777  1.504535
##   11     2.948974  0.4148744  1.577500
##   12     3.051209  0.4190821  1.607100
##   13     3.121594  0.4116678  1.633030
##   14     3.224966  0.4081584  1.661069
##   15     3.223102  0.4093197  1.653023
##   16     3.137133  0.4084612  1.635706
##   17     3.033415  0.4125177  1.596948
##   18     3.056691  0.4120972  1.606322
##   19     3.088082  0.4098343  1.613294
##   20     3.050203  0.4051016  1.608124
##
## Rsquared was used to select the optimal model using the largest value.
## The final value used for the model was ncomp = 4.
```

```
plot(plsTune)
```



```
plsTune$results |>
  filter(ncomp == 4)
```

```
##   ncomp     RMSE  Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 1     4 1.634346 0.5127511 1.158204 1.175888  0.1763008 0.3701309
```

In the PLS model, 4 components was the optimal model with an R^2 value of .576.

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?

```
plsPred <- predict(plsTune, newdata = X_test)
postResample(pred = plsPred, obs = y_test)
```

```
##      RMSE  Rsquared       MAE
## 1.1044103 0.5996569 0.8792745
```

The measurement statistics from the test set are similar to the training set indicating that the model is predicting values about the same which makes it a good model.

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

```
top10 <- varImp(plsTune)$importance |>
  arrange(desc(Overall)) |>
  head(10)
```

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

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

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

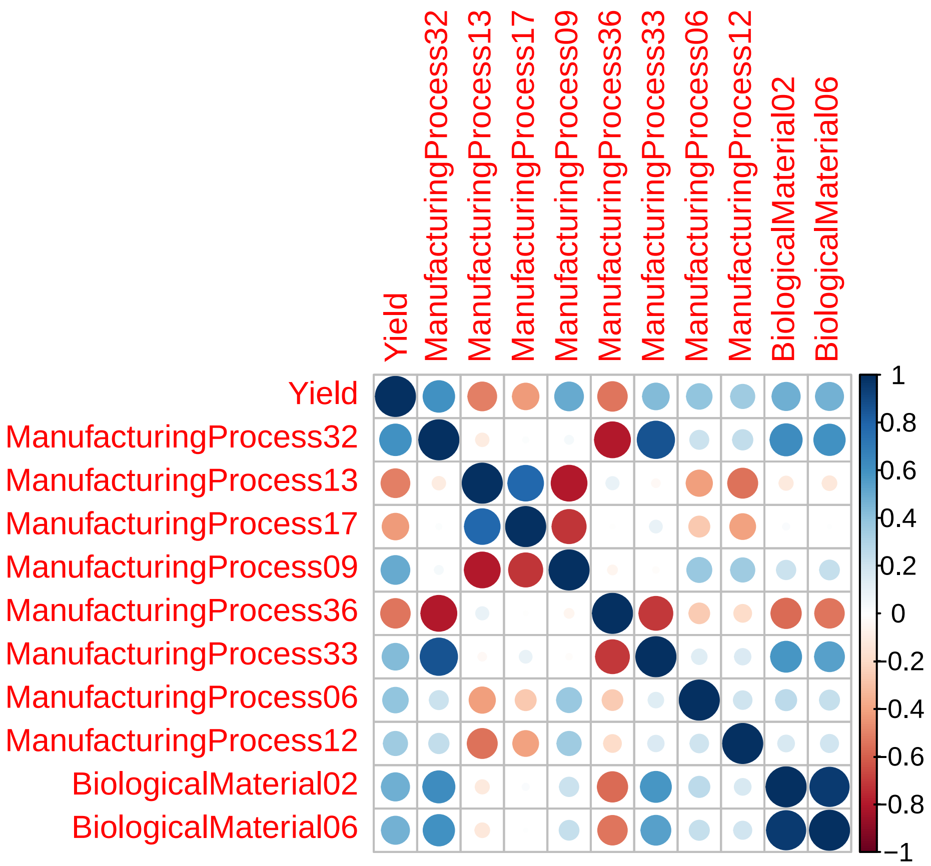
```
top10
```

```
##                          Overall
## ManufacturingProcess32 100.00000
## ManufacturingProcess13  89.99177
## ManufacturingProcess17  83.95322
## ManufacturingProcess09  83.21071
## ManufacturingProcess36  76.64583
## ManufacturingProcess33  64.34552
## ManufacturingProcess06  63.76321
## ManufacturingProcess12  59.03898
## BiologicalMaterial02    57.57342
## BiologicalMaterial06    56.92581
```

Looking at the top 10 predictors and the weights of their importance, the manufacturing process predictors have the most importance.

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

```
imputed |>
  select(c('Yield', row.names(top10))) |>
  cor() |>
  corrplot::corrplot()
```



It's important to understand which predictors improve the yield and which will decrease it. Manufacturing processes 13, 17, and 36 reduce the yield, while the others improve it. Additionally, recognizing which predictors are correlated to each other, may also help in finding yield improvements.