# Homework 05: Applied Predictive Modeling

Dongzhe Li

104434089

Exercise 6.1

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

*library (caret)*

*data(tecator)*

1. **In this example the predictors are the measurements at the individual frequencies. Because the frequencies lie in a systematic order (850–1,050nm), the predictors have a high degree of correlation. Hence, the data lie in a smaller dimension than the total number of predictors (215). Use PCA to determine the effective dimension of these data. What is the effective dimension?**

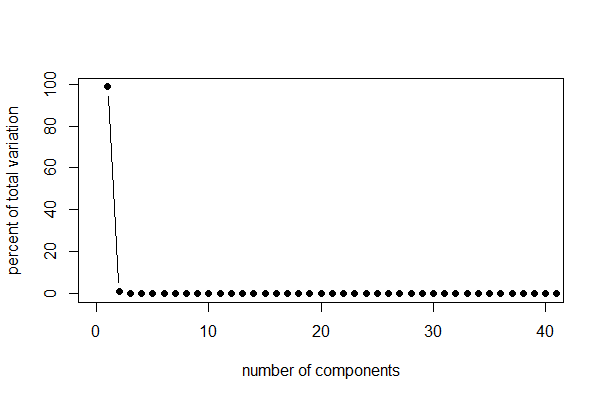
First, apply PCA to the data

*pc=prcomp(absorp, scale = TRUE)*

Then we can get the variance of each component and calculate the percentage of each variance. The first 6 percentages of variance are listed below:

98.626192582 0.969705229 0.279324276

0.114429868 0.006460911 0.002624591



From the figure we can see that the first component account for most of the information, so there is only one effective dimension based on the linear analysis.

1. **Split the data into a training and a test set, pre-process the data, and build each variety of models described in this chapter. For those models with tuning parameters, what are the optimal values of the tuning parameter(s)?**

The training and test set is built below

*Training=createDataPartition(endpoints[, 3], p = 0.75, list= FALSE)*

*absorbtrain=absorp[Training,]*

*absorbtest=absorp[-Training,]*

*proteintrain =endpoints[ Training, 3]*

*proteintest = endpoints[-Training,3]*

In this case we only choose the third column of endpoints----the protein as the predictor. Build the linear model

*lm=train(x = absorbtrain, y = proteintrain, method = "lm", trControl = trainControl(method = "repeatedcv", repeats = 5))*

The summary is printed below:

Linear Regression

163 samples

100 predictors

No pre-processing

Resampling: Cross-Validated (10 fold, repeated 5 times)

Summary of sample sizes: 147, 147, 146, 147, 147, 147, ...

Resampling results

RMSE Rsquared RMSE SD Rsquared SD

1.52 0.819 0.781 0.175

The RMSE is 1.52

The Robust linear regression "rlm" model is built below, however, rlm does not allow the covariance matrix of the predictors to be singular to ensure that predictors are not singular, we will pre-process the predictors using PCA

*rlm = train( x=absorbtrain, y=proteintrain, method="rlm", preProcess=c("pca"), trControl=trainControl(method="repeatedcv",repeats=5) )*

Robust Linear Model

163 samples

100 predictors

Pre-processing: principal component signal extraction, scaled, centered

Resampling: Cross-Validated (10 fold, repeated 5 times)

Summary of sample sizes: 146, 148, 147, 146, 147, 147, ...

Resampling results

RMSE Rsquared RMSE SD Rsquared SD

2.6 0.275 0.349 0.17

The PLS model:

Partial Least Squares

163 samples

100 predictors

Pre-processing: centered, scaled

Resampling: Cross-Validated (10 fold, repeated 5 times)

Summary of sample sizes: 146, 148, 147, 146, 147, 147, ...

Resampling results across tuning parameters:

ncomp RMSE Rsquared RMSE SD Rsquared SD

1 2.93 0.0975 0.274 0.115

2 2.29 0.44 0.333 0.163

3 1.75 0.664 0.404 0.167

4 1.61 0.712 0.401 0.159

5 1.18 0.854 0.177 0.0467

6 1.11 0.871 0.166 0.0437

7 1.08 0.876 0.163 0.048

8 0.95 0.906 0.167 0.0412

9 0.911 0.915 0.167 0.0389

10 0.86 0.924 0.166 0.0329

11 0.792 0.935 0.136 0.0252

12 0.733 0.947 0.154 0.0226

13 0.715 0.949 0.176 0.0236

14 0.694 0.953 0.159 0.0233

15 0.72 0.95 0.174 0.0233

16 0.804 0.934 0.277 0.0448

17 0.876 0.916 0.371 0.0818

18 0.927 0.902 0.425 0.101

19 0.95 0.894 0.472 0.117

20 0.941 0.894 0.483 0.121

21 0.921 0.899 0.459 0.112

22 1 0.877 0.566 0.161

23 1.04 0.869 0.595 0.172

24 1.04 0.872 0.564 0.152

25 1.08 0.862 0.593 0.165

RMSE was used to select the optimal model using the smallest value.

The final value used for the model was ncomp = 14.

And the PCR :

Principal Component Analysis

163 samples

100 predictors

No pre-processing

Resampling: Cross-Validated (10 fold, repeated 5 times)

Summary of sample sizes: 145, 147, 148, 147, 147, 147, ...

Resampling results across tuning parameters:

ncomp RMSE Rsquared RMSE SD Rsquared SD

1 2.94 0.106 0.274 0.113

2 2.68 0.234 0.273 0.165

3 2.3 0.441 0.369 0.18

4 1.65 0.697 0.43 0.176

5 1.47 0.766 0.286 0.102

6 1.13 0.869 0.174 0.0413

7 1.13 0.868 0.174 0.0408

8 1.12 0.868 0.168 0.04

9 1.1 0.876 0.147 0.0333

10 1.02 0.898 0.158 0.0328

11 1.05 0.891 0.174 0.0371

12 0.889 0.923 0.117 0.0231

13 0.795 0.937 0.145 0.0233

14 0.765 0.942 0.139 0.0228

15 0.744 0.944 0.137 0.0211

16 0.684 0.953 0.136 0.0214

17 0.691 0.953 0.142 0.0213

18 0.721 0.951 0.159 0.0208

19 0.718 0.951 0.18 0.0252

20 0.722 0.95 0.19 0.0265

21 0.782 0.94 0.247 0.0373

22 0.819 0.932 0.296 0.0551

23 0.853 0.925 0.324 0.0594

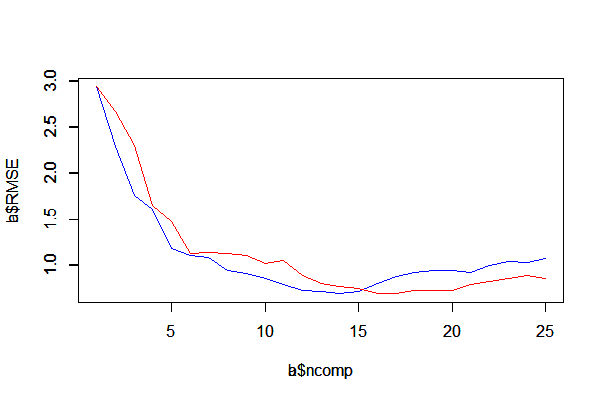
24 0.882 0.918 0.362 0.0683

25 0.853 0.922 0.341 0.07

RMSE was used to select the optimal model using the smallest value.

The final value used for the model was ncomp = 16.

These two models have similar RMSE but the PLS requires less components.

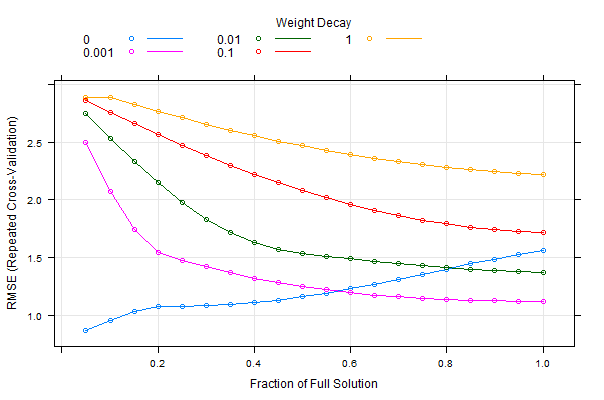


PCR

PLS

At last we build the elastic net model:

*ENet = train(x = absorbtrain, y = proteintrain,method = "enet",trControl = trainControl(method="repeatedcv",repeats=5), preProcess = c("center", "scale"),tuneGrid = expand.grid(lambda = c(0, .001, .01, .1, 1), fraction = seq(0.05, 1, length = 20)))*



1. **Which model has the best predictive ability? Is any model significantly better or worse than the others?**

From the RMSE result of different model, we find that PLS model has better performance since it is especially suited to handling highly correlated data. And linear model has the worst performance overall.

1. **Explain which model you would use for predicting the fat content of a sample.**

PLS model can be used to predict the fat content as predicting protein in the previous problems.

Execise 6.2

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

*library (AppliedPredictiveModeling)*

*data (permeability)*

1. **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?**

*zero = nearZeroVar( fingerprints )*

*fingerprints = fingerprints[,-zero]*

There are 719 near-zero variance fingerprints, leaving 388 left for modeling

There are 165\*1107=182655 elements in the previous matrix and after removing 0, there are only 165\*388=64020 elements.

1. **Split the data into a training and a test set, pre-process the data, and tune a partial least squares model. How many latent variables are optimal and what is the corresponding resampled estimate of R2?**

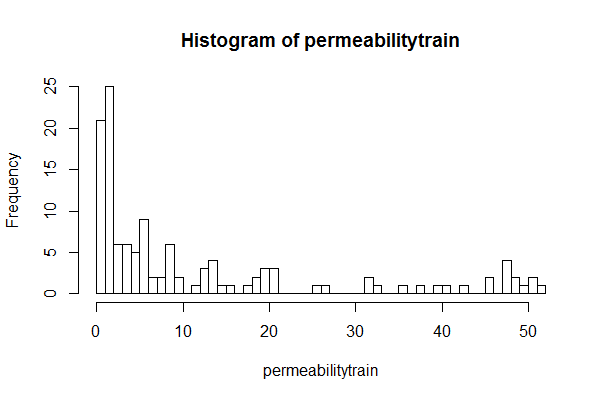
*Training = createDataPartition( permeability, p=0.75 )*

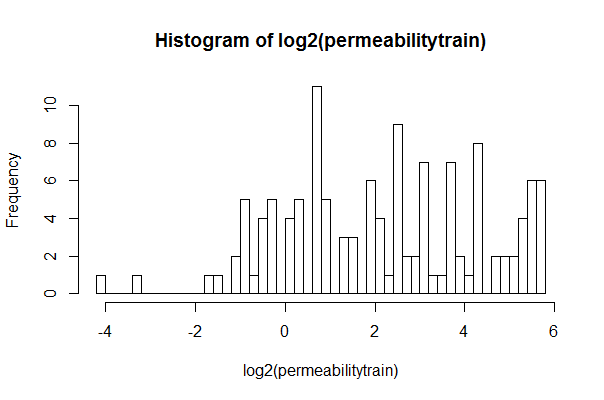
Set p=0.75 and

*fingerprintstrain <- fingerprintsnonzero [Training$Resample1,]*

*permeabilitytrain <- permeability[Training$Resample1,]*

Create the training dataset for PLS model, however, the permeability is not a symmetric distribution. We also need to pre-process the permeability before building the model





I choose log2 to modify the data and build the model.

The result of PLS model (tunelength=25) is listed below:

Partial Least Squares

125 samples

388 predictors

No pre-processing

Resampling: Repeated Train/Test Splits Estimated (25 reps, 0.75%)

Summary of sample sizes: 96, 96, 96, 96, 96, 96, ...

Resampling results across tuning parameters:

ncomp RMSE Rsquared RMSE SD Rsquared SD

1 1.93 0.281 0.272 0.145

2 1.71 0.436 0.302 0.142

3 1.66 0.471 0.279 0.123

4 1.67 0.475 0.29 0.119

5 1.68 0.478 0.251 0.111

6 1.69 0.485 0.272 0.114

7 1.68 0.493 0.276 0.116

8 1.65 0.513 0.266 0.11

9 1.63 0.528 0.245 0.101

10 1.64 0.537 0.233 0.0949

11 1.69 0.523 0.239 0.0982

12 1.72 0.513 0.237 0.0943

13 1.75 0.503 0.246 0.104

14 1.81 0.479 0.243 0.108

15 1.84 0.471 0.252 0.112

RMSE was used to select the optimal model using the smallest value.

The final value used for the model was ncomp = 9.

when n=10, the Rsquared reach the maximum value 0.537

1. **Predict the response for the test set. What is the test set estimate of R2?**

*ypredict = predict( plsTune, newdata=fingerprintstest )*

And the R-squared is 0.56 which is not very far from the real value.

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

For different models, I test lm, rlm, PCR, Enet.

LM:

*lm = train( fingerprintstrain, permeabilitytrain, method="lm", preProcess=c("center","scale"), trControl=trainControl(method="repeatedcv",repeats=5) )*

Linear Regression

125 samples

388 predictors

Pre-processing: centered, scaled

Resampling: Cross-Validated (10 fold, repeated 5 times)

Summary of sample sizes: 113, 111, 112, 112, 113, 113, ...

Resampling results

RMSE Rsquared RMSE SD Rsquared SD

32.5 0.192 11.1 0.193

rlm

Robust Linear Model

125 samples

388 predictors

Pre-processing: principal component signal extraction, scaled, centered

Resampling: Cross-Validated (10 fold, repeated 5 times)

Summary of sample sizes: 113, 111, 112, 112, 113, 113, ...

Resampling results

RMSE Rsquared RMSE SD Rsquared SD

11.7 0.517 3.57 0.235

PCR

Principal Component Analysis

125 samples

388 predictors

Pre-processing: principal component signal extraction, scaled, centered

Resampling: Repeated Train/Test Splits Estimated (25 reps, 0.75%)

Summary of sample sizes: 96, 96, 96, 96, 96, 96, ...

Resampling results across tuning parameters:

ncomp RMSE Rsquared RMSE SD Rsquared SD

1 16 0.0599 1.47 0.0787

2 16 0.0597 1.47 0.0785

3 14.8 0.196 0.0232 0.229

4 12.5 0.401 0.445 0.164

5 11.9 0.466 0.652 0.127

6 11.9 0.461 0.657 0.127

7 12.3 0.416 0.538 0.125

8 12.3 0.427 0.791 0.132

9 12.7 0.393 0.246 0.177

10 12.4 0.432 0.697 0.125

11 12 0.47 0.293 0.153

12 11.9 0.471 0.238 0.152

13 12 0.463 0.257 0.152

14 12.1 0.458 0.0867 0.164

15 12 0.461 0.0297 0.168

16 12.3 0.44 0.14 0.163

17 12.2 0.462 0.0875 0.129

18 11.9 0.461 0.144 0.155

19 11.9 0.457 0.289 0.14

20 11.9 0.461 0.127 0.146

21 11.6 0.481 0.374 0.18

22 11.5 0.486 0.315 0.175

23 11.5 0.488 0.328 0.178

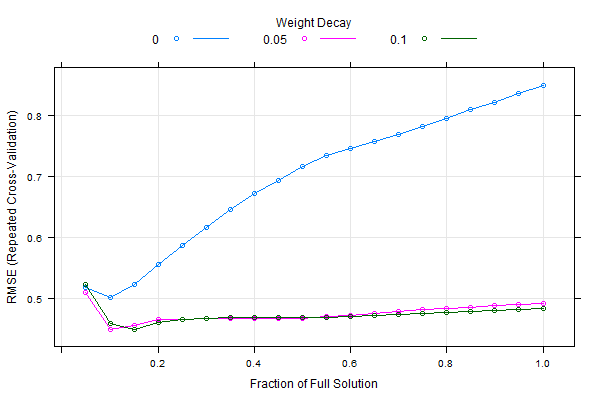
24 11.6 0.493 0.326 0.143

25 12 0.48 0.18 0.157

RMSE was used to select the optimal model using the smallest value.

The final value used for the model was ncomp = 23

Enet

****

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

From the result above, the elastic net solution the best penalized model choice for this problem. It has the lowest RMSE value.

## Appendix

Q1

library(caret)

data(tecator)

#calculate the mean and variance of each column of endpoints

apply(absorp, 2, mean)

apply(absorp, 2, var)

pc=prcomp(absorp, scale = TRUE)

print( pc$center )

print( pc$scale )

biplot(pc, scale = 0)

#calculate the largest 6 eigenvalues percentage

pc\_var = pc$sdev^2

pve = pc\_var/sum(pc\_var)\*100

head(pve,10)

plot( pve, xlim=c(0,40), type='b', pch=16, xlab='number of components', ylab='percent of total variation' )

# 98.626192582 0.969705229 0.279324276 0.114429868 0.006460911 0.002624591

#only the first dimension is effective

#we use the third attribute in the endpoints

set.seed(0)

Training=createDataPartition(endpoints[, 3], p = 0.75, list= FALSE)

absorbtrain=absorp[Training,]

absorbtest=absorp[-Training,]

proteintrain =endpoints[ Training, 3]

proteintest = endpoints[-Training,3]

control = trainControl(method = "repeatedcv", repeats = 5)

#linear model

set.seed(1)

lm=train(x = absorbtrain, y = proteintrain, method = "lm", trControl = trainControl(method = "repeatedcv", repeats = 5))

mean(proteintrain)

#rlm

rlm = train( x = absorbtrain, y = proteintrain, method="rlm", preProcess=c("pca"), trControl=trainControl(method="repeatedcv",repeats=5) )

#pcr

set.seed(1)

PCR=train(x = absorbtrain, y = proteintrain, method = "pcr", preProcess=c("pca"),trControl = control, tuneLength = 25)

#pls

set.seed(1)

PLS=train(x = absorbtrain, y = proteintrain, method = "pls",trControl=trainControl(method="repeatedcv",repeats=5), preProcess = c("center", "scale"),tuneLength = 25)

#PCR

PCR <- train(x = absorbtrain, y = proteintrain,method = "pcr", trControl = trainControl(method="repeatedcv",repeats=5), tuneLength = 25)

comps <- rbind(PLS$results, PCR$results)

comps$Model <- rep(c("PLS", "PCR"), each = 25)

a=comps[1:25,1:6]

b=comps[26:50,1:6]

plot(a$ncomp,a$RMSE,col="blue",type="l")

par(new=TRUE)

plot(b$ncomp,b$RMSE,col="red",type="l")

legend(col=c("red","blue"))

set.seed(1)

#enet

ENet = train(x = absorbtrain, y = proteintrain,method = "enet",trControl = trainControl(method="repeatedcv",repeats=5), preProcess = c("center", "scale"),tuneGrid = expand.grid(lambda = c(0, .001, .01, .1, 1), fraction = seq(0.05, 1, length = 20)))

plot(ENet)

Q2

library(caret)

library(AppliedPredictiveModeling)

data(permeability)

zero = nearZeroVar( fingerprints )

fingerprintsnonzero = fingerprints[,-zero]

Training = createDataPartition( permeability, p=0.75 )

fingerprintstrain = fingerprintsnonzero [Training$Resample1,]

permeabilitytrain =permeability[Training$Resample1,]

fingerprintstest=fingerprintsnonzero [-Training$Resample1,]

permeabilitytest=permeability[-Training$Resample1,]

hist(permeabilitytrain,breaks=50)

hist(log2(permeabilitytrain),breaks=50)

set.seed(3)

control=trainControl(method = "LGOCV")

plsTune =train(x = fingerprintstrain, y =log10(permeabilitytrain) , method = "pls",tuneGrid = expand.grid(ncomp = 1:15),trControl = control)

plsTune

ypredict = predict( plsTune, newdata=fingerprintstest )

rsquared\_pls = cor(ypredict,permeabilitytest,method="pearson")^2

set.seed(3)

lm = train( fingerprintstrain, permeabilitytrain, method="lm", preProcess=c("center","scale"), trControl=trainControl(method="repeatedcv",repeats=5) )

ypredict\_lm = predict( lm, newdata=fingerprintstest )

r2\_lm = cor(ypredict\_lm,permeabilitytest,method="pearson")^2

set.seed(3)

rlm = train( fingerprintstrain, permeabilitytrain, method="rlm", preProcess=c("pca"), trControl=trainControl(method="repeatedcv",repeats=5) )

ypredict\_rlm = predict( rlm, newdata=fingerprintstest )

r2\_rlm = cor(ypredict\_rlm,permeabilitytest,method="pearson")^2

set.seed(3)

PCR=train(fingerprintstrain, permeabilitytrain, method = "pcr", preProcess=c("pca"),trControl = control, tuneLength = 25)

set.seed(3)

enet = train( fingerprintstrain, permeabilitytrain, method="enet", tuneGrid = expand.grid(lambda = c(0, .05, .1), fraction = seq(0.05, 1, length = 20)), trControl=trainControl(method="repeatedcv",repeats=5) )

plot(enet)