P8106 Data Science Homework 1

Emil Hafeez

Contents

EMIL: DO YOU NEED TO CV ON THE TEST DATASET TOO? FOR ALL PROBLEMS DOES RIDGE REGRESSION REQUIRE THE X PASSED IN TO BE THE MATRIX SHE STANDARDIZES IN L5RMD?

There were missing values in resampled performance measures. for ridge reg

DO RMSE AT THE END USING RESAMPLE

Section 1: Setup

Libraries and Options

```
train_data = read_csv("./data/solubility_train.csv")
test_data = read_csv("./data/solubility_test.csv")
train_data = na.omit(train_data)
test_data = na.omit(test_data)
set.seed(1993)
```

Prompt

In this exercise, we will predict solubility of compounds using their chemical structures.

The training data are in the file “solubility train.csv” and the test data are in “solubil- ity test.csv”. Among the 228 predictors, 208 are binary variables that indicate the presence

or absence of a particular chemical substructure, 16 are count features, such as the number of bonds or the number of bromine atoms, and 4 are continuous features, such as molecular weight or surface area. The response is in the column “Solubility” (the last column).

Questions

(A) Fit a linear model using least squares on the training data and calculate the mean squared error using the test data.

(B) Fit a ridge regression model on the training data, with chosen by cross-validation. Report the test error.

(C) Fit a lasso model on the training data, with chosen by cross-validation. Report the test error and the number of non-zero coefficient estimates in your model.

(D) Fit a principle component regression model on the training data, with M chosen by cross-validation. Report the test error and the value of M selected by cross-validation.

(E) Which model will you choose for predicting solubility?

Section 2: Implementation and Results

Problem A)

The dataframe consists of 229 variables. There are a large number of predictors and potential collinearity (implying large variance), but a linear model is implemented to start. While it would therefore be computationally unwise to compute all or best subsets and choose that way, this approach is included for sake of completeness

Note that in performing cross-validation, you want to apply CV both when finding the predictors with the largest correlation with the response; and, you want to do it when fitting the model using only those predictors.

```
# summary(train_data)
```

```
regsubsets_obj = regsubsets(Solubility ~ ., data = train_data, method = "backward", nbest = 1)
# summary(regsubsetsObj)
plot(regsubsets_obj, scale = "bic")
```

We train the model using cross-validation.

```
set.seed(1993)
control_cv <- trainControl(method = "repeatedcv", number = 10, repeats = 5)

# lm(Solubility ~ ., data = train_data)

fit.lm <- train(Solubility~.,
          data = train_data,
          method = "lm",
          trControl = control_cv)
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading
```

```
# fit.lm$finalModel
```

Then, we apply the model to the test data and obtain the MSE for evaluation purposes.

```
#Fitting training model on test set
pred.lm = predict(fit.lm, newdata = test_data)
rmse_lm = RMSE(pred.lm, test_data$Solubility)
rmse_lm
```

```
## [1] 0.7455802
```

Problem B)

(B) Fit a ridge regression model on the training data, with chosen by cross-validation. Report the test error.

Since regularization methods like ridge regression is not scale-invariant like LS methods (since ridge regression minimizes the objective function which includes a penalty term) so we must standardize our predictors first.

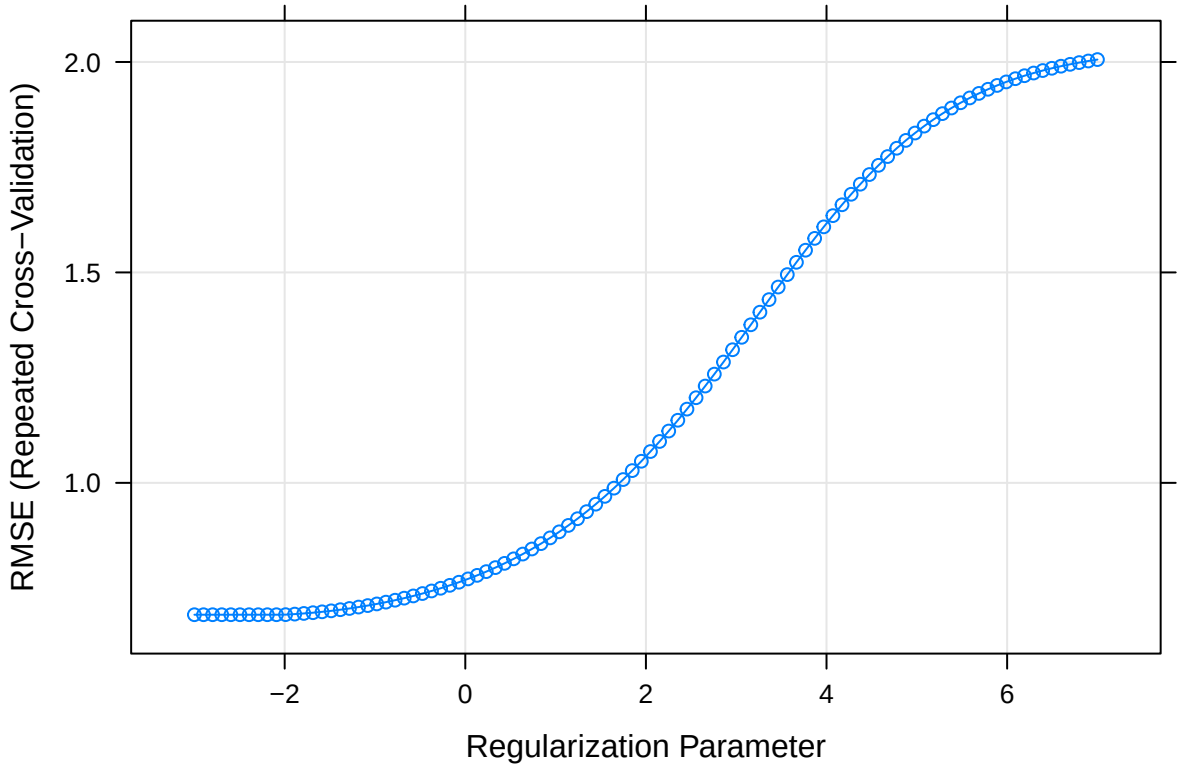
```
train_data_model_matrix <- model.matrix(Solubility ~ ., train_data)[ ,-1]

control_ridge <- trainControl(method = "repeatedcv", number = 10, repeats = 5)

set.seed(1993)
```

```
ridge.fit <- train(train_data_model_matrix, train_data$Solubility,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = 0,
                                          lambda = exp(seq(7, -3, length=100))),
                   # preProc = c(scale), glmnet does this by default
                   trControl = control_ridge)

plot(ridge.fit, xTrans = log)
```




```
ridge.fit$bestTune
```

```
##    alpha    lambda
## 10     0 0.1235747
```

```
# coefficients in the final model
coef(ridge.fit$finalModel, s = ridge.fit$bestTune$lambda)
```

```
## 229 x 1 sparse Matrix of class "dgCMatrix"
##                               1
## (Intercept)        7.6079945976
## FP001              0.0365731685
## FP002              0.1297827520
## FP003             -0.0746147741
## FP004             -0.1735299445
```

```
## FP005            -0.0269922335
## FP006            -0.0988361626
## FP007             0.0355961926
## FP008             0.0328429577
## FP009            -0.0243063334
## FP010             0.0381761924
## FP011             0.1049137871
## FP012            -0.0660812861
## FP013            -0.0858488096
## FP014             0.0074244405
## FP015            -0.0276478247
## FP016            -0.1176341295
## FP017            -0.1888430780
## FP018            -0.1564230119
## FP019             0.0189674055
## FP020             0.1261960935
## FP021             0.0098663343
## FP022             0.1523963079
## FP023            -0.1632821265
## FP024            -0.1272129778
## FP025             0.0152232152
## FP026             0.1638033471
## FP027             0.2228231252
## FP028             0.0730210830
## FP029            -0.0555734124
## FP030            -0.1647623464
## FP031             0.1953369715
## FP032            -0.1688264777
## FP033             0.3110143965
## FP034            -0.1772888302
## FP035            -0.1363319464
## FP036            -0.0082056554
## FP037             0.2715095751
## FP038             0.1387109382
## FP039            -0.4188302598
## FP040             0.4679367335
## FP041            -0.0305322717
## FP042             0.0429158457
## FP043             0.0650064129
## FP044            -0.3155887642
## FP045             0.1191310588
## FP046             0.1045294243
## FP047            -0.0511206873
## FP048             0.0353648044
## FP049             0.3042316000
## FP050            -0.0963063229
## FP051            -0.0288117314
## FP052            -0.0990260491
## FP053             0.3241978423
## FP054            -0.0884626247
## FP055            -0.1718357277
## FP056            -0.0670541303
## FP057            -0.1161771571
## FP058             0.0470282759
```

```
## FP059            -0.3186181919
## FP060             0.1029874491
## FP061            -0.0735925356
## FP062             0.0790195200
## FP063             0.2235272241
## FP064             0.1503605396
## FP065            -0.1604637648
## FP066             0.1283349662
## FP067            -0.0957431214
## FP068             0.0735236273
## FP069             0.1494930515
## FP070            -0.1949273065
## FP071             0.1178346736
## FP072             0.2938050866
## FP073            -0.1010918560
## FP074             0.1300749497
## FP075             0.2359363505
## FP076             0.0343169632
## FP077             0.1005329917
## FP078            -0.0362003621
## FP079             0.1549097911
## FP080             0.1112517434
## FP081            -0.1722497506
## FP082             0.1033016478
## FP083            -0.2628451661
## FP084             0.1940151724
## FP085            -0.2944051729
## FP086            -0.0691145675
## FP087             0.0508880717
## FP088             0.1433671733
## FP089            -0.1455277036
## FP090            -0.0402155131
## FP091             0.0488738466
## FP092             0.0263239911
## FP093             0.1198732537
## FP094            -0.1075414544
## FP095             0.0356942517
## FP096             0.0509303017
## FP097             0.1131267736
## FP098            -0.0566705842
## FP099             0.1499069886
## FP100            -0.0746650876
## FP101             0.0973129509
## FP102             0.1313771437
## FP103            -0.1338040630
## FP104            -0.1254119124
## FP105            -0.0800800587
## FP106             0.0078638842
## FP107            -0.0140275443
## FP108             0.0470663573
## FP109             0.2516111030
## FP110             0.0309587471
## FP111            -0.3715059662
## FP112            -0.1080423907
```

```
## FP113              0.0874931305
## FP114              0.1200596980
## FP115              0.1231060132
## FP116              0.1544905654
## FP117             -0.1038671032
## FP118             -0.1355231882
## FP119              0.2165494188
## FP120             -0.0630735305
## FP121             -0.1018770940
## FP122              0.1934141601
## FP123             -0.0378185760
## FP124              0.2303054291
## FP125              0.0887977031
## FP126             -0.3083752188
## FP127             -0.3020366995
## FP128             -0.2231001570
## FP129              0.0550703660
## FP130             -0.2529667803
## FP131              0.2762456566
## FP132             -0.0476868595
## FP133             -0.1336278488
## FP134             -0.1103668162
## FP135              0.1758693361
## FP136              0.1010049874
## FP137              0.0820727631
## FP138              0.1785829167
## FP139              0.0421934682
## FP140              0.0851338627
## FP141             -0.1481109054
## FP142              0.3516627358
## FP143              0.3161206686
## FP144              0.0299086470
## FP145             -0.2032067541
## FP146             -0.0909625443
## FP147              0.2399892147
## FP148              0.0064686076
## FP149             -0.0242899342
## FP150              0.1356166423
## FP151              0.1203091571
## FP152             -0.0182527819
## FP153             -0.1029555872
## FP154             -0.3521183298
## FP155              0.1301685028
## FP156             -0.2304999617
## FP157              0.0325926763
## FP158             -0.0410564403
## FP159              0.2597844590
## FP160             -0.1335618141
## FP161             -0.0630642478
## FP162              0.0630419369
## FP163              0.2251357775
## FP164              0.2520266849
## FP165             -0.0339119567
## FP166              0.0972832824
```

```
## FP167             -0.1400575478
## FP168             -0.1295925945
## FP169             -0.1660604151
## FP170              0.0597296537
## FP171              0.2389791767
## FP172             -0.3229796897
## FP173              0.3249242184
## FP174             -0.1204221821
## FP175             -0.0437119468
## FP176              0.3316890771
## FP177             -0.0109751860
## FP178             -0.0116277767
## FP179              0.0796239183
## FP180             -0.1143663141
## FP181              0.0998827191
## FP182             -0.0465102917
## FP183             -0.0147596147
## FP184              0.2744136048
## FP185             -0.0709527620
## FP186             -0.2348358018
## FP187              0.1455451329
## FP188              0.2442763844
## FP189              0.0494734515
## FP190              0.2012337381
## FP191              0.0719663681
## FP192              0.0741646048
## FP193             -0.0738990029
## FP194              0.0328861654
## FP195             -0.0415575312
## FP196              0.0833037456
## FP197             -0.0004699364
## FP198              0.2088558536
## FP199             -0.0120701862
## FP200             -0.0875781978
## FP201             -0.2778320987
## FP202              0.2467538132
## FP203              0.0584619715
## FP204             -0.0823972015
## FP205             -0.1142230177
## FP206             -0.0967385308
## FP207             -0.0400239160
## FP208              0.0057122503
## MolWeight         -1.1151876599
## NumAtoms          -0.4056107101
## NumNonHAtoms      -0.6466174682
## NumBonds          -0.3496803663
## NumNonHBonds      -0.3314374539
## NumMultBonds      -0.0945513574
## NumRotBonds       -0.1945882755
## NumDblBonds       -0.0311888707
## NumAromaticBonds  -0.1148407353
## NumHydrogen        0.0843773653
## NumCarbon         -0.2629728334
## NumNitrogen        0.3948773129
```

```
## NumOxygen           0.3644667619
## NumSulfer          -0.5859066258
## NumChlorine        -0.6501256934
## NumHalogen         -0.4932219927
## NumRings           -0.2648692283
## HydrophilicFactor   0.1350830470
## SurfaceArea1        0.0805512872
## SurfaceArea2        0.0489015916
```

The optimal value of lambda chosen by repeated cross-validation is 0.1235747.