Practicing Linear, Ridge, Lasso, PC Regression models

Hana Akbarnejad

2/27/2020

In this exercise, we will predict solubility of compounds using their chemical structures. We will use traing and test data, fit linear, Ridge, Lasso, and principle component regression models and identify the best mosel that can be used for predicting solubility.

After importing data and omitting the missing values, we should start by defining test and train model matrices for predictors and also define the responses in train and test data. We will later need them for our models in this exercise. I also defined a *train_control* that uses the repeated cross validation method, with 50 resamples (5 times, 10 each).

```
set.seed(1)

# setting up x matrix and y for train and test data
x_train = model.matrix(solubility ~ ., train_data)[,-1]
y_train = train_data$solubility

x_test = model.matrix(solubility ~ ., test_data)[,-1]
y_test = test_data$solubility

# need to define a train control, reampling method specified as repeated Cv
# tuneGrid: candidates for tuning parameter
# I chose to pre-process!
# "train" here is resampling method and is different from the next train :)
# I will use it every where needed!

train_control = trainControl(method = "repeatedcv", number = 10, repeats = 5)
```

(a)

In the first part, we are interested in fitting a linear model using least squares on the training data and calculate the mean square error using the test data.

```
set.seed(1)

# fit lm model using training data
lm_fit = train(x_train, y_train,
               method = "lm",
               trControl = train_control)

# predict model using training model and test data
test_pred = predict(lm_fit, test_data)
summary(test_pred)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -9.804  -3.941  -2.537  -2.792  -1.353   1.408
```

```
# calculating MSE using training and test models
# mean((y_pred - y_test)^2)
mse = mean((test_pred - test_data$solubility)^2)
```

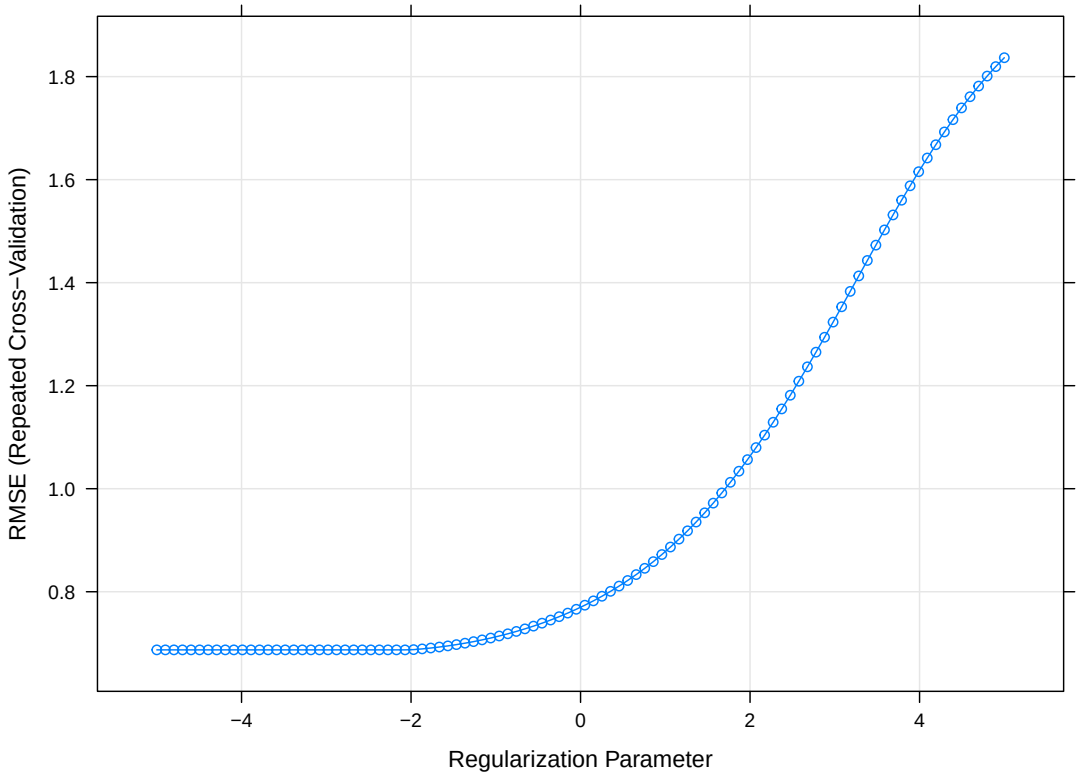
After fitting a linear model and using it to predict the test data,we obtain mean square error of 0.556.

(b)

In this part, we want to choose λ and fit a ridge regression model on the training data and find out the test error.

We first fit a ridge regression model. We use cross validation to choose the optimal value of λ. We use $\alpha = 0$ which is the ridge penalty to obtain λ using *cv.glmnet* function. Alternatively, we can Use *train()* function from *caret* package to fit Ridge regression:

```
set.seed(1)

ridge_fit = train(x_train, y_train,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = 0,
                                         lambda = exp(seq(-5, 5, length = 100))),
            preProc = c("center", "scale"),
            trControl = train_control)

# plot the parameters over RMSE to choose the best option
plot(ridge_fit, xTrans = function(x) log(x))
```




```
# best lambda!
ridge_lambda = ridge_fit$bestTune

# summary of coefficients in Ridge regression
# coef(ridge_fit$finalModel,ridge_fit$bestTune$lambda)

# MSE from Ridge
pred_ridge_fit = predict(ridge_fit, newdata = test_data)
ridge_mse = mean((pred_ridge_fit - y_test)^2)
```

Uisng Ridge regression, we can see that the optimum λ chosen is 0.126.

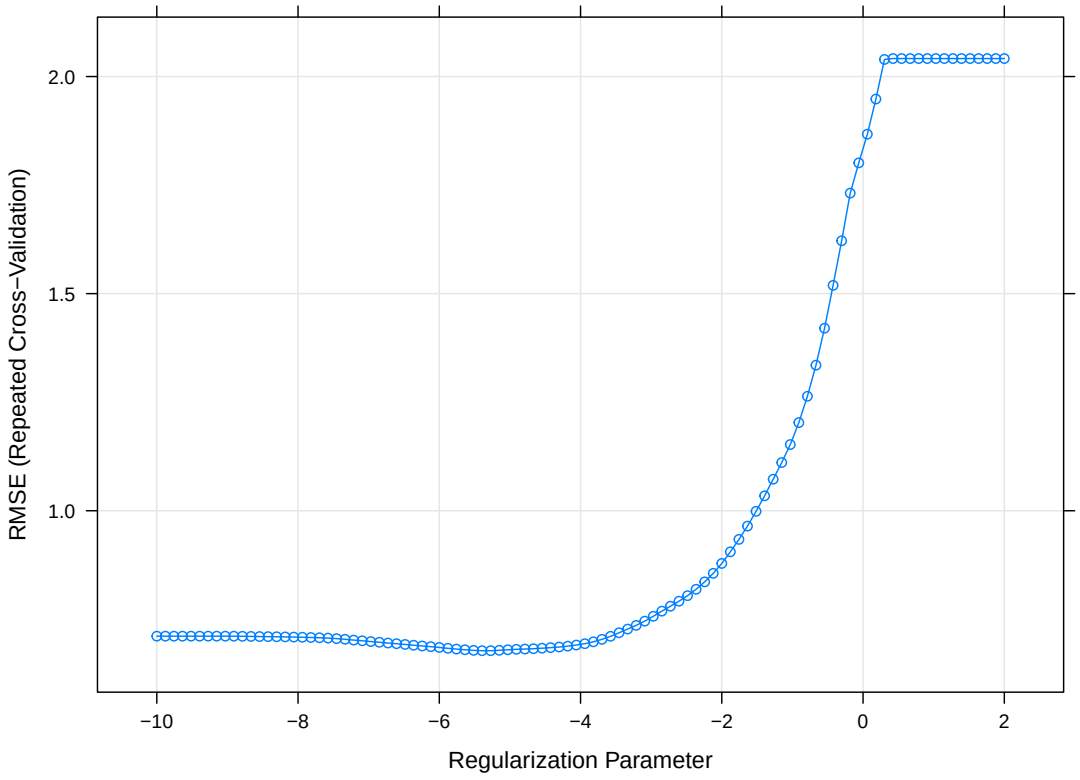
To compute the test error, we need to predict the test data using the model we have built and then use that to compute the MSE. We can see that the test error obtained from this model is 0.513.

(c)

In this part, we want to choose λ by cross-validation and fit a Lasso regression model on the training data and find the test error.

```
set.seed(1)
# fitting lasso regression, we should set alpha as 1
lasso_fit = train(x_train, y_train,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = 1,
                                         lambda = exp(seq(-10, 2, length = 100))),
```

```
            preProc = c("center", "scale"),
            trControl = train_control)

# plotting
plot(lasso_fit, xTrans = function(x) log(x))
```



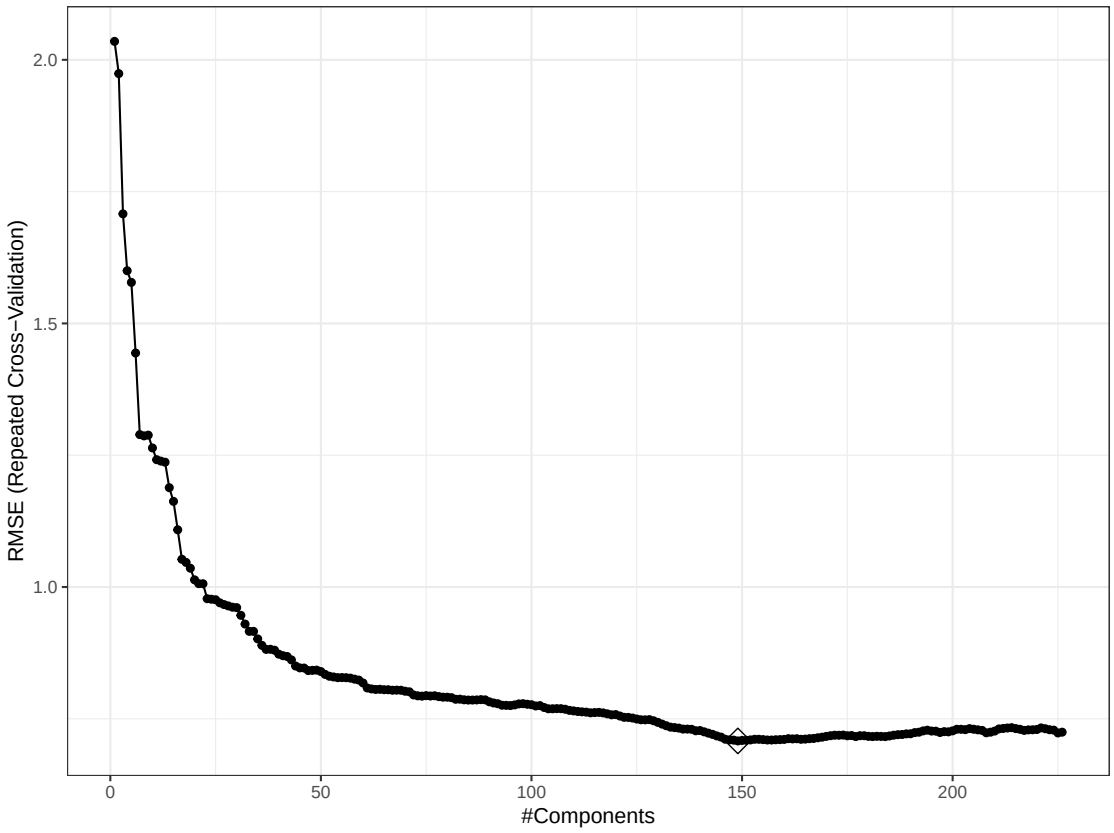

```
# best lambda!
lasso_lambda = lasso_fit$bestTune

# summary of coefficients in Lasso regression
lasso_coeff = coef(lasso_fit$finalModel,lasso_fit$bestTune$lambda)

# MSE from Lasso
pred_lasso_fit = predict(lasso_fit, newdata = test_data)
lasso_mse = mean((pred_lasso_fit - y_test)^2)

# non-zero coefficients
coeff_df = as.data.frame(as.matrix(lasso_coeff))

nonzero_coeff = coeff_df %>%
  filter(coeff_df[1] != 0) %>%
  nrow()
```

Using the fitted Lasso model, cross validation and the plot above that depicts RMSE against tuning parameters, We can see that the λ obtained from cross validation is 0.005 using $\alpha = 1$, the test error is 0.499, and that the number of non-zero parameters is 148.

(d)

In this part, we want to choose M by cross-validation, and fit a principle component regression model on the training data. We furthur want to compute the test error:

```
set.seed(1)

# fitting PCR
pcr_fit = train(x_train, y_train, method = "pcr",
                tuneGrid = data.frame(ncomp = 1:226), # it should be from 1 to the number of predictors
                trControl = train_control,            # wired results when I knit, so we try to omit th
                preProc = c("center", "scale"))

trans = preProcess(x_train, method = c("center", "scale"))

pred_pcr = predict(pcr_fit$finalModel, newdata = predict(trans, x_test),
                   ncomp = pcr_fit$bestTune$ncomp)

pcr_mse = mean((y_test - pred_pcr)^2)

pcr_m = pcr_fit$bestTune

ggplot(pcr_fit, highlight = TRUE) + theme_bw()
```



We can observe that using principal components regression, the value of M chosen by cross validation is 149 which can also be observed on the plot above, and the test error is 0.541.

(e)

The above analysis show that using seed 1 and on this computer(!), MSE obtained from Lasso Regression is 0.499, MSE from Ridge regression is 0.513 and the MSE from PCR is 0.541. The MSE from the linear regression is the highest among these four models with the value of 0.556.

The (α, λ) parameters from Ridge and Lasso were (0, 0.126) and (1, 0.005), respectively, and the M from PCR is 149.

lambda for lasso much lower than ridge (0.004 vs 0.126).

(f)

```
resamp = resamples(list(pcr = pcr_fit, lasso = lasso_fit, ridge = ridge_fit, lm = lm_fit))
summary(resamp) # comment on rmse
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: pcr, lasso, ridge, lm
## Number of resamples: 50
##
## MAE
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## pcr   0.4493418 0.5021588 0.5417937 0.5427523 0.5736116 0.7111992    0
## lasso 0.4084059 0.4903347 0.5147265 0.5176255 0.5504138 0.6639900    0
## ridge 0.4200226 0.4827439 0.5202268 0.5236104 0.5618375 0.6971025    0
## lm    0.4277377 0.4915503 0.5249318 0.5310003 0.5530262 0.7069382    0
##
## RMSE
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## pcr   0.6015178 0.6554806 0.7016990 0.7080427 0.7574392 0.9014038    0
## lasso 0.5365094 0.6346469 0.6718319 0.6776365 0.7288339 0.8266256    0
## ridge 0.5657252 0.6400041 0.6739751 0.6871419 0.7442585 0.8830344    0
## lm    0.5748467 0.6639700 0.7100120 0.7115004 0.7445387 0.8860734    0
##
## Rsquared
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## pcr   0.8238483 0.8687718 0.8818169 0.8815807 0.8990229 0.9200157    0
## lasso 0.8508578 0.8753876 0.8948457 0.8910048 0.9044047 0.9273178    0
## ridge 0.8299172 0.8698131 0.8884934 0.8878594 0.9035499 0.9245791    0
## lm    0.8208162 0.8630037 0.8851720 0.8809016 0.9009296 0.9202780    0
```

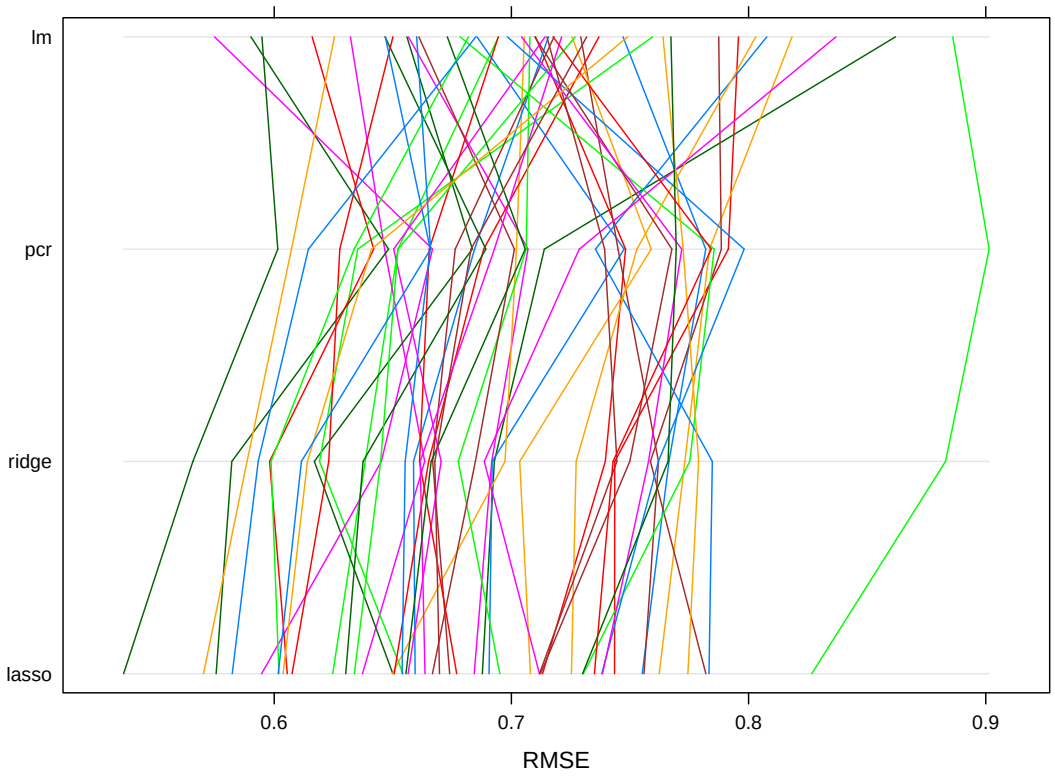
In this section we want to select one model which is the best between linear, Ridge, Lasso, and PC regressions that can be used for predicting solubility. For this purpose, we review MSE values discissed in previous part and see that deciding between these models, Lasso offers the best model with the minimum error and that the linear regression is not complex enough to cover our datapoints in a proper way.

we apply *resamples()* function on these models and look at the mean in *RMSE* results. We can see that RMSE for linear model is the higher with the value of 0.711, the second highest mean RMSE is for the PCR model with 0.707, then it is Ridge with 0.687 and finally Lasso, which has the lowest RMSE with the value

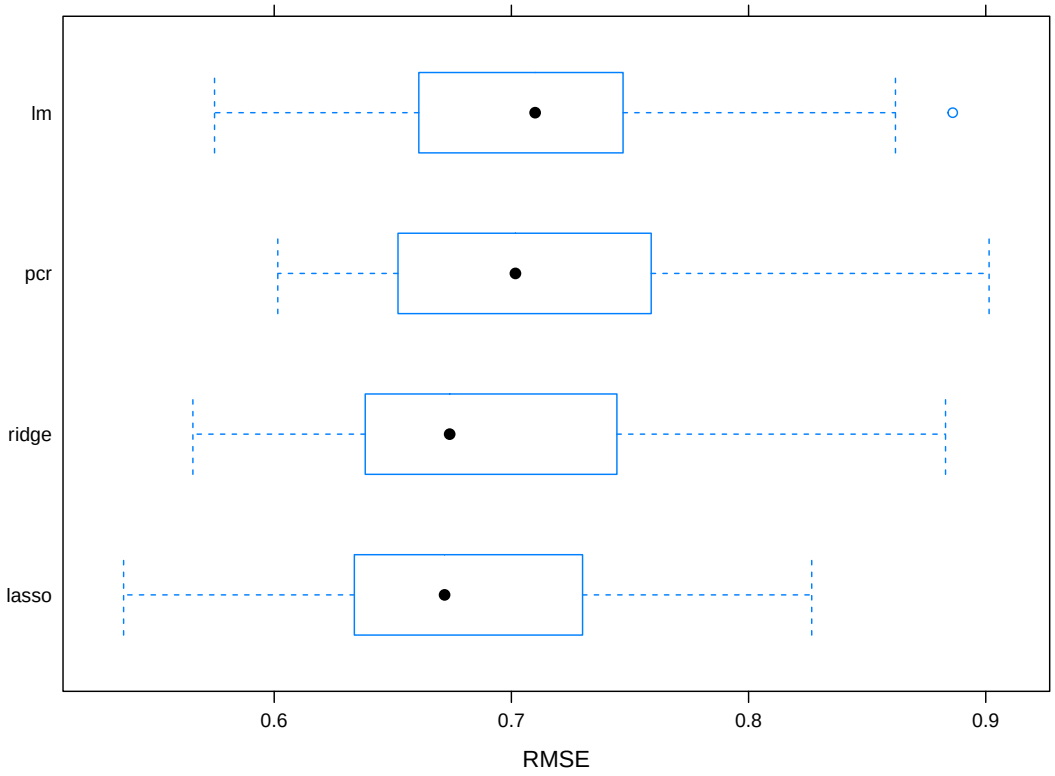
of 0.678. These results confirm that Lasso is a better fit to our data compared to the other three models and is the model I would choose for analyzing this dataset to predict the solubility.

Below, we can also see a parallel plot and boxplots of the methods we used that can help visualize the results we discussed.

```
parallelplot(resamp, metric = "RMSE")
```




```
bwplot(resamp, metric = "RMSE")
```


lm
pcr
ridge
lasso
0.6
0.7
0.8
0.9
RMSE