Toronto

Regularization: LASSO and Ridge

Soni Manan[a]

[a] *College of Professional Studies, Master of Professional Studies in Analytics.*

Subject: *ALY6015* ***NUID***: *002982645*

Under the guidance of

Dr. Prof. Alex Maizlish

Introduction

Dataset was selected from library (ISLR) in R and dataset has data of colleges of United States. This was all regarding Private-Profit based and Public colleges of United states in 1995, that is it provided.

Regularization: LASSO and Ridge?

- Regularization is the method which is used to adjust the model to increase accuracy. It used to calibrate model's cost function by addition of a term.
- Terms in LASSO and Ridge are almost same which is shown below.

$$\textbf{LASSO:}\ \lambda\Sigma|\beta_j|$$
$$\textbf{RIDGE:}\ \lambda\Sigma\beta_j^2$$

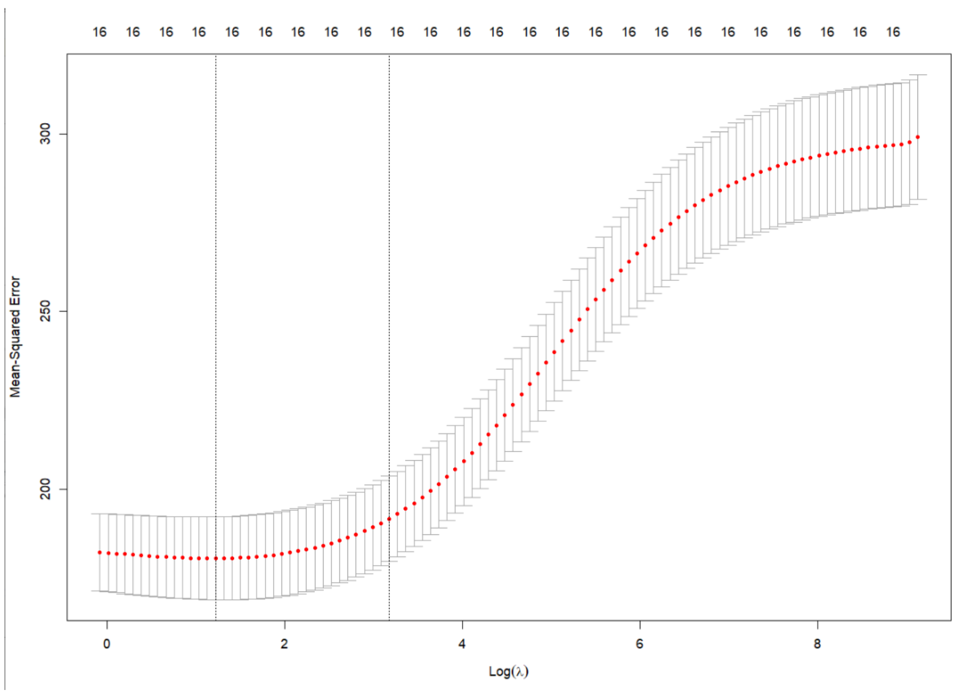
- Beta (β) is basically difference between value of sample and predictor of that sample. In Lasso mod had taken whereas in Ridge it squared.
- Usage of LASSO and RIDGE. In nutshell, when sample size is small or number of predictor variable is less go for LASSO because it reduces difference between performance and variance and that's the difference which we identify through Lambda (λ). It's basically penalty for making wrong choices That's how it calibrates the model. For ridge its vice and versa.
- So from lambda you can guess and minimize the error.
- Mean Squared Error (MSE) is the way to calculate accuracy of model. So, these model helps to improve by introducing bias to the variance which leads to good MSE.
- The less MSE the less error but both methods can reduce MSE to certain value.
- Drawback of both methods that plots are difficult to interpret as multiple variables incepts with 0.

Train-Test Split for Model

We used to split the samples for training and testing the model. The threshold is 70% and 30% respectively.

Ridge Model

- Ridge model is to find that coefficient (Lambda) that lowers the RSS (Sum of squared residuals).
- In R, `cv.glmnet()` is used to perform this and It does standardization itself.
- This will return that lambda.
- And This will be the plot for the college dataset.
- Lambda min for this is 3.402
- It uses k-fold cross-validation and `cv.glmnet()` will take k=10 by default.
- We need to set a parameter `alpha` which is for choosing between LASSO and Ridge.
- For Ridge, its 0 and 1 for LASSO.



- This plot is hard to interpret but the second dotted line is around 3 as we can notice. So lowest point of the curve is the value of lambda.
- That red dotted line is of cross-validation with bounds of standard-deviation.
- Both dotted lines gives minimum mean cross-validation error.

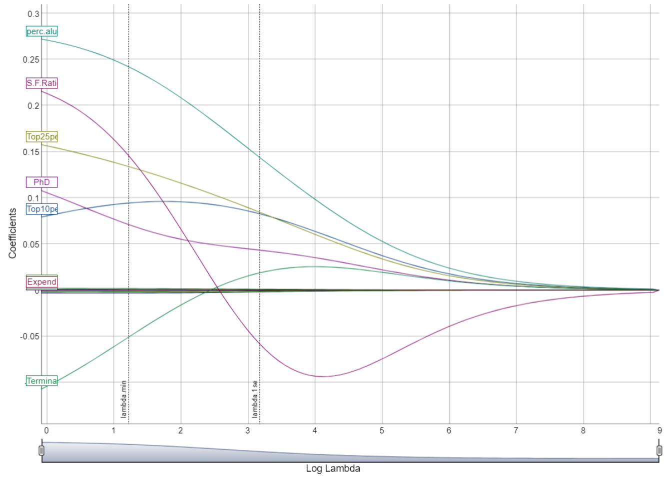
```
> model$lambda.min
```

- This piece of code gives value of lambda.

```
Measure: Mean-Squared Error

    Lambda Index Measure    SE Nonzero
min  3.402    86   180.4 11.69      16
1se 24.002    65   191.6 12.05      16
```

- Model performs or tests multiple lambda values and gives best of it. For lambda 0, model has no effect of methods and for some value > 0 its effective then for bigger value it goes to overfitting.



- This plot shows that at some point and by lambda, the variables are impact on variables. And at last they merge to 0.
- There is lambda.1se is should be calculated for most regulated model. To calculate this, `glmnet()` replace `lambda.min` to `lambda.1se` and for future scope lambda.1se is better.

```
(Intercept) 44.5342080556
Apps         0.0001578535
Accept       0.0001388454
Enroll      -0.0002024842
Top10perc    0.0827326800
Top25perc    0.0844394200
F.Undergrad -0.0000987657
P.Undergrad -0.0008091728
Outstate     0.0004597768
Room.Board   0.0011642983
Books       -0.0019169669
Personal    -0.0016284069
PhD          0.0433812353
Terminal     0.0187858852
S.F.Ratio   -0.0582134034
perc.alumni  0.1437457567
Expend       0.0001063201
```

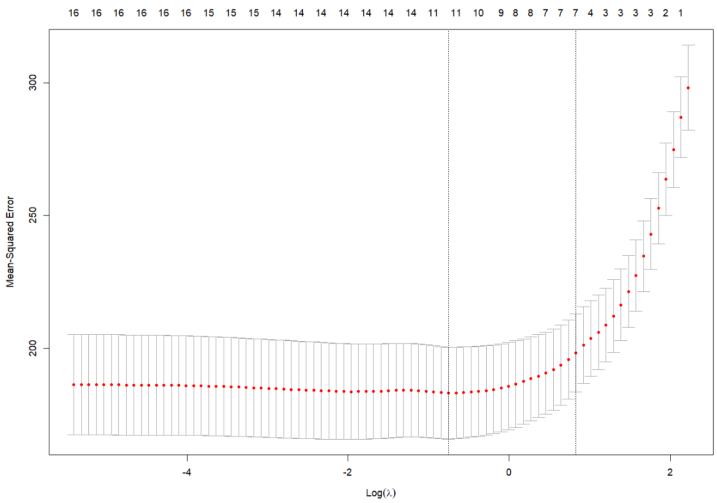
- This is the train RMSE value and test RMSE is respectively 13.707 and 12.530. Train RMSE is greater than Test RMSE. RMSE is basically Root mean square error, and its standard deviation of the residuals.

v

```
[1] 13.70734
> test.rmse
[1] 12.53098
```

Lasso Model

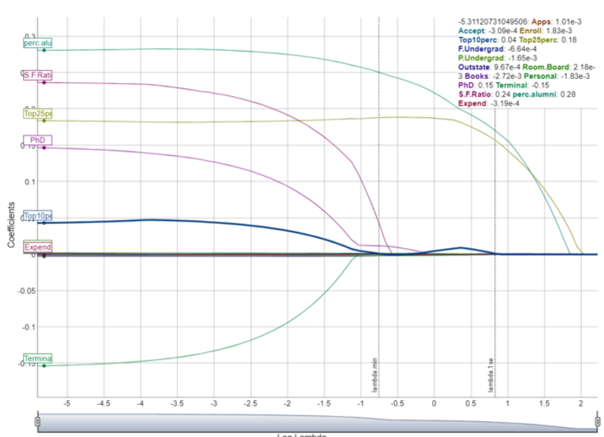
- LASSO is used when samples size is less as it used to eliminate variables unlike Ridge. We'll see that later.
- Noticing as the two dotted lines, curve started to incline to 300, so that's the value of lambda.
- The indicators at the top of the plot, those number are of non-zero regression coefficients, which increasing from 1 to 16.



- As we discussed earlier, difference in this plot is , LASSO is tries to eliminate the errors.
- But its most effective when samples are limited or variables are less.
- Looking at value of lambda.min and lambda.1se. Accuracy actually improved at 1se not at min. Comparing both these plots, 1se seems to be good instead min.
- Alpha will be 1 for this test.

```
Measure: Mean-Squared Error

    Lambda Index Measure    SE Nonzero
min 0.4712    33   183.1 17.30      11
1se 2.2911    16   198.3 14.69       7
```



- The non-valued in this diagram are eliminated by LASSO. Intercept is reduced compared to RIDGE.

```
17 x 1 sparse Matrix of class "dgCMatrix"
                        s0
(Intercept) 41.8934731931
Apps         .
Accept       .
Enroll       .
Top10perc    0.0013204377
Top25perc    0.1572508591
F.Undergrad  .
P.Undergrad -0.0002299048
Outstate     0.0010119658
Room.Board   0.0003069530
Books        .
Personal    -0.0003918499
PhD          .
Terminal     .
S.F.Ratio    .
perc.alumni  0.1706600239
Expend       .
```

- The trained RMSE is 13.877 and test RMSE is 12.507. There is no particular threshold but RMSE of test < RMSE of train it could be underfitting for both methods.
- In this case both RMSE is almost similar.

Conclusion

By finishing this assignment, I understood the idea of regularisation. How to test if the version is under fitted or overfitting and what may be achieved in both of the cases. The essential distinction among Ridge and Lasso changed into additionally understood.

Both the methods, resulted same over here as RMSE of test was less than RMSE of train.

References

Utkarsh Kumar, 1st January 2020, Create Matrix and Data frame from lists in R Programming. *Source*: https://www.geeksforgeeks.org/create-matrix-and-data-frame-from-lists-in-r-programming

Zach, August 26 2021, When to Use Ridge & Lasso Regression *Source*: https://www.statology.org/how-to-calculate-mse-in-r/

Zach, November 11 2020, Introduction to Ridge Regression *Source*: https://www.statology.org/how-to-calculate-mse-in-r/

Appendix

pacman::p_load(tidyverse, rio, party, Metrics, ggplot2, dplyr, vcd, webr, Matrics, ggridges, simex, Hmisc, plotrix, gginference,corrplot,ggplot2,ggpubr,BSDA,glmnet,coefplot,ISLR,caret)

```
#Importing Dataset
View(College)


#Split the data into a train and test set – refer to the Feature_Selection_R.pdf document for information on how to split a dataset.
set.seed(123)
IndexSet <- sample(2, nrow(College),
        replace = T,
        prob = c(0.7,0.3))
datatrain <- College[IndexSet == 1,]
setdata <- College[IndexSet == 2,]

#Removing private variable
College = College[,-1]
view(College)
datatrain = datatrain[,-1]

setdata = setdata[,-1]
view(datatrain)

train_x <-model.matrix(Grad.Rate~.,datatrain)[,-1]
test_x <- model.matrix(Grad.Rate~.,setdata)[,-1]
train_x2 <- datatrain$Grad.Rate
test_x2 <- setdata$Grad.Rate

#Ridge
set.seed(123)
glmRidge <- cv.glmnet(train_x, train_x2, nfolds = 10, alpha = 0)
glmRidge
plot(glmRidge)
coefpath(glmRidge)

best_lambda <- log(glmRidge$lambda.min)
best_lambda

simp_lambda <- log(glmRidge$lambda.1se)
simp_lambda

# Determine the performance of the fit model against the training set by calculating the root mean square error (RMSE). sqrt(mean((actual - predicted)^2))
simp_model <- glmnet(train_x, train_x2, alpha = 0, lambda = glmRidge$lambda.1se)
coef(simp_model)
simp_model
```

# Determine the performance of the fit model against the training set by calculating the root mean square error (RMSE). sqrt(mean((actual - predicted)^2))
pred_train <- predict(simp_model1 , newx = train_x)
train.rmse <-rmse(train_x2, pred_train)
train.rmse

# Determine the performance of the fit model against the test set by calculating the root mean square error (RMSE). Is your model overfit?
pred_test <- predict(simp_model1 , newx = test_x)
test.rmse <-rmse(test_x2, pred_test)
test.rmse

# Lasso
set.seed(123)
glmLasso <- cv.glmnet(train_x, train_x2, nfolds = 10, alpha = 1)
glmLasso
plot(glmLasso)

best_lambda <- log(glmLasso$lambda.min)
best_lambda

simp_lambda <- log(glmLasso$lambda.1se)
simp_lambda

#Fit a LASSO regression model against the training set and report on the coefficients. Do any coefficients reduce to zero? If so, which ones?
simp_model <- glmnet(train_x, train_x2, alpha = 1, lambda = glmLasso$lambda.1se)
coef(simp_model)
simp_model

coefpath(glmLasso)

# Determine the performance of the fit model against the training set by calculating the root mean square error (RMSE). sqrt(mean((actual - predicted)^2))
pred_train <- predict(simp_model , newx = train_x)
train.rmse <-rmse(train_x2, pred_train)
train.rmse

# Determine the performance of the fit model against the test set by calculating the root mean square error (RMSE). Is your model overfit?
pred_test <- predict(simp_model , newx = test_x)
test.rmse <-rmse(test_x2, pred_test)
test.rmse

# Stepwise selection method
step(lm(Grad.Rate ~ ., data = College), direction = 'both')
model_step <- step(lm(Grad.Rate ~ ., data = College), direction = 'both')
summary(model_step)