APPLICATION OF GROUP LASSO

EXST 7999 Independent Study Report

Christopher Kuetsinya

April 20, 2022

Introduction

In this age of big data, the number of features measured on a person or object can be large, and might be larger than the number of observations. In most cases, only a few number of features are relevant in measuring the response variable. Hence, one may use a sparse statistical model to analyze these kinds of data. A sparse statistical model is one having only a small number of nonzero parameters or weights. It represents a classic case of "less is more" and a sparse model can be much easier to estimate.

Lasso

The Lasso is one of the sparse statistical methods. It combines the least-square loss with a penalty known as the l_1 *penalty*. This penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter lambda is sufficiently large. Hence, much like best subset selection, the lasso performs variable selection.

Group Lasso

Generalizations of the Lasso method has also been made to cater for data with structurally grouped features. In such cases, it is ideal to select or omit all the coefficients within a group together. The group lasso is a method tailored to handle such cases and is achieved by using sums of (unsquared) l_2 penalties.

In this report, I analyzed a data built to have features that are structurally grouped, with the Lasso and Group Lasso methods. After analyzing, their performance metrics were compared to assess which method works best in a situation where features are structurally grouped.

Data Description

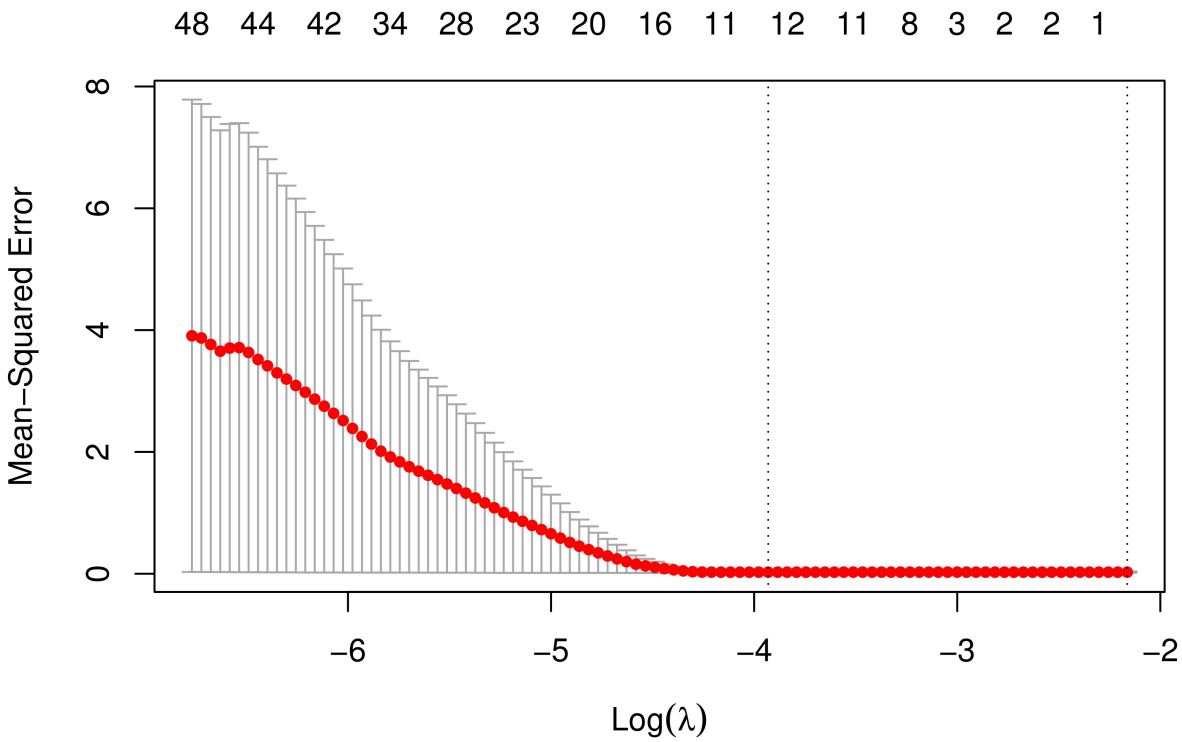
The data used is a Gene expression data from the microarray experiments of mammalian eye tissue samples. This data contains 120 samples, 100 predictors and a numeric response variable. The 100 predictors are made up of expression levels expanded from 20 filtered genes using 5 basis B-splines; as a result each 5 consecutive predictors are a grouped gene. The response variable gives expression level of gene TRIM32, which causes Bardet-Biedl syndrome.

Data Analysis

The data was split, 80 for training and 40 for testing. A Lasso and group Lasso models were built on the training data. Correlation between the fitted values and the test data response variable, R-squared and Root Mean Square Error were the metrics used in comparing the performance of the two models.

Lasso

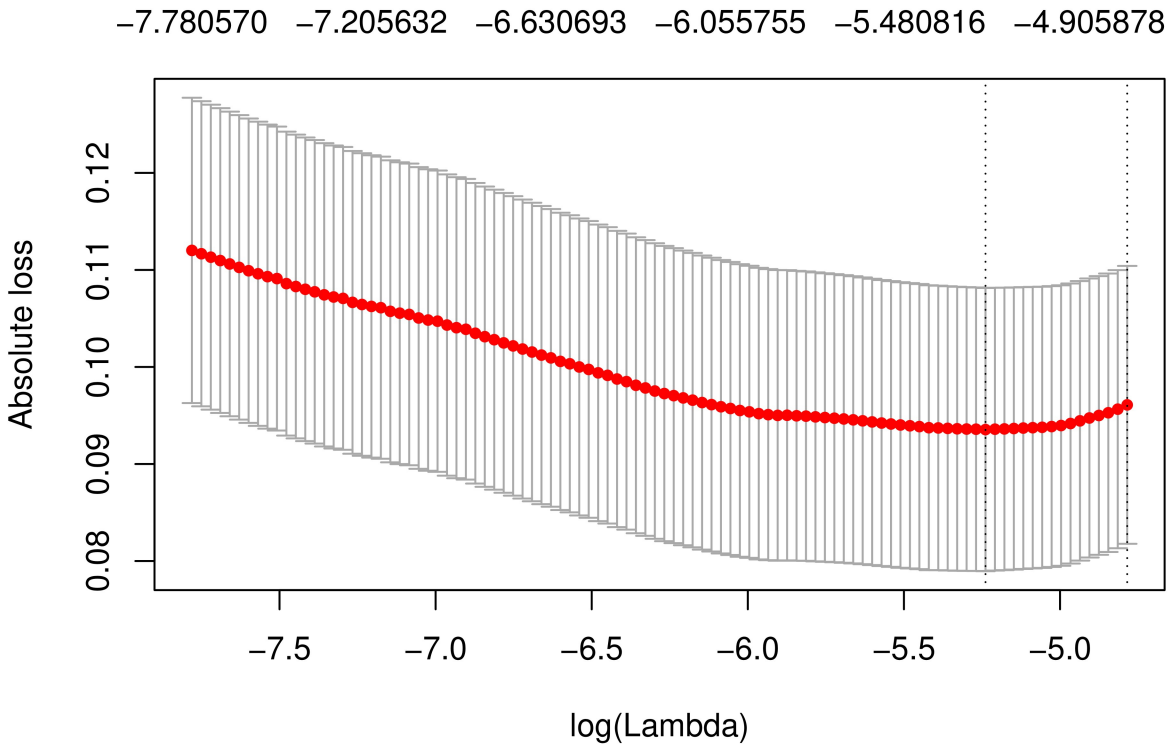
Considering the dimensionality of the data, a lasso model would make the coefficients of some of the variables shrink to zero. Using the ***glmnet*** package in R, a 5-fold cross validation was used to select the optimal lambda.



The optimal lambda obtained was 0.003. A total of 12 were selected in this model, hence the coefficients of 188 variables were shrunk to zero.

Group Lasso

Now, since the variables in the data have been made to have some structural grouping, a group lasso model was built to make the coefficients of non-essential groups zero and compared its performance metrics to that of the lasso method.
With the use of the ***gglasso*** package in R, the grouping structure was introduce and a 5-fold cross validation was used to select the optimal lambda with the smallest absolute loss.



The optimal lambda obtained was 0.003. A total of 3 groups were selected in this model, hence 3 groups had non-zero coefficients which means 15 predictors were selected in the group lasso model.

Performance Metrics

After building the models with the training data, predictions were made for both models using the test data. The predicted values obtained were used in calculating the correlation, R-squared and Root MSE.

Table 1: Model performance metrics

Model	r	R^2	$RMSE$
Lasso	0.75	0.454	0.013
Group Lasso	0.86	0.716	0.007

Findings & Discussion

The correlation metric checks the relationship between the predicted values using the test data and the response variable of the test set. Thus, the higher the better. Also, the R-squared represents the amount of variability explained in the response by the predictors. For this metric also, the higher the better. The RMSE defines the standard deviation of the residuals (prediction errors). Residuals are a measure of how far

from the regression line data points are; RMSE is a measure of how spread out these residuals are. Hence, it is ideal to have a relatively small RMSE. Comparing all three model performance metrics, the group lasso was better.

Conclusion

The group lasso can be applied in both regression and classification cases. It is ideal to use when features have grouping structure or the data include some dummy variables that are used to code a multilevel categorical predictor. Based on the example given in this report, group lasso works better when a data has features with natural grouping structure.

References

- Hastie, T., Tibshirani, R., & WainWright, M. (2018). Statistical Learning with Sparsity: The Lasso and Generalizations. Boca Raton: CRC Press.

- Yang, Y., & Yang, Y., & Zou, H. (2015). A fast unified algorithm for solving group-lasso penalize learning problems. Statistics and Computing, 25(6), 1129-1141.Zou, H. (2015). A fast unified algorithm for solving group-lasso penalize learning problems. Statistics and Computing, 25(6), 1129-1141.

- Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). The elements of statistical learning: data mining, inference, and prediction (Vol. 2, pp. 1-758). New York: springer.
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, p. 18). New York: springer.

Appendix

```
library(gglasso)
library(glmnet)
data(bardet)
N=120
n=80
set.seed(2)
id=sample(1:N,size=n,replace=F)
X1 = bardet$x[id,]
Y1 = (as.matrix(bardet$y))[id,]
X2 = bardet$x[-id,]
Y2 = (as.matrix(bardet$y))[-id,]
summary.output <- function(y,pred){
  #correlation
  cr<-cor(y,pred)
  #R-squared
  r2<-1-mean((y-pred)^2)/mean((y-mean(y))^2)
  #RMSE
  rmse<-sqrt(mean((y-pred))^2)
  c(cr,r2,rmse)
}
```

```r
# LASSO
lasso <- cv.glmnet(x=X1,y=Y1)
plot(lasso)
lasso$lambda.min
lasso.pred<-as.vector(predict(lasso,X2,s="lambda.min"))
Lmin=round(summary.output(Y2,lasso.pred), digits = 3)
```

```r
# GROUP LASSO
group1 <- rep(1:20,each=5)
grouplasso <- cv.gglasso(x=bardet$x,y=bardet$y,group=group1,pred.loss="L1")
plot(grouplasso)
grouplasso$lambda.min
grouplasso.pred<-as.vector(predict(grouplasso,X2,s="lambda.min"))
GLmin=round(summary.output(Y2,grouplasso.pred), digits = 3)
```

```r
# MODEL COMPARISON
rbind(Lmin,GLmin)
```

```
## 101 x 2 sparse Matrix of class "dgCMatrix"
##                       s1             1
## (Intercept)  8.418694289  8.3542881563
## V1           .            .
## V2           .            .
## V3           .            .
## V4           .            .
## V5          -0.059710187  .
## V6           .            .
## V7           .            .
## V8           .            .
## V9           .            .
## V10          .            .
## V11          .            .
## V12          .            .
## V13          .            .
## V14          .            .
## V15          .            .
## V16          .            0.0019194854
## V17          .            0.0022094278
## V18          .           -0.0013249771
## V19          .           -0.0001646422
## V20         -0.007296621 -0.0045429489
## V21          .            0.0809268194
## V22          .            0.0813762213
## V23         -0.043253562 -0.0571297790
## V24          .           -0.0158744616
## V25         -0.561531508 -0.1782331794
## V26          .            0.0056015507
## V27          .            0.0123943241
## V28          .           -0.0010006368
## V29          .           -0.0001803526
## V30         -0.234019405 -0.0188351028
## V31          .            .
## V32          .            .
```

```
## V33          .            .
## V34          .            .
## V35          .            .
## V36          0.069341859  .
## V37          .            .
## V38          .            .
## V39          .            .
## V40          .            .
## V41          .            .
## V42          .            .
## V43          .            .
## V44          0.016016319  .
## V45          .            .
## V46         -0.174851408  .
## V47          .            .
## V48          .            .
## V49          .            .
## V50          .            .
## V51          .            .
## V52         -0.038938278  .
## V53          .            .
## V54          0.065799780  .
## V55          .            .
## V56          .            .
## V57          .            .
## V58          .            .
## V59          .            .
## V60          0.014360663  .
## V61          .            .
## V62          .            .
## V63          .            .
## V64          .            .
## V65          0.103446495  .
## V66          .            .
## V67          .            .
## V68          .            .
## V69          .            .
## V70          .            .
## V71          .            .
## V72          .            .
## V73          .            .
## V74          .            .
## V75          .            .
## V76          .            .
## V77          .            .
## V78          .            .
## V79          .            .
## V80          .            .
## V81          .            .
## V82          .            .
## V83          .            .
## V84          .            .
## V85          .            .
## V86          .            .
```

```
## V87           .             .
## V88           .             .
## V89           .             .
## V90           .             .
## V91           .             .
## V92           .             .
## V93           .             .
## V94           .             .
## V95           .             .
## V96           .             .
## V97           .             .
## V98           .             .
## V99           .             .
## V100          .             .
```