Northeastern University
College *of* Professional Studies

Project 3

Regularization Practices

Dickson Wanjau

ALY6015 81058 Intermediate Analytics SPRING 2019 CPS

Instructor: Dr. Roseanna Hopper

Due Date: May 1, 2019.

Introduction

In regression analysis, overfitting occurs when the model fits the training data too well, capturing all the noises (unexplained variations in data or error terms). In this case, we notice a high accuracy in the training dataset, whereas the same model will result in a low accuracy on the test dataset. This means the model has fitted the line so well to the train dataset that it failed to generalize it to fit well on an unseen dataset. Thus, the predictions from the model are highly variable, leading to unstable results.

Regularization is a technique used to control overfitting by adding a penalty term to the cost function on the number of parameters in the model. The model will be less likely to fit the noise of the training data and will improve the generalization abilities of the model.

There are 3 types of regularization techniques: L1 Regularization (LASSO), L2 Regularization (Ridge) and L1/L2 regularization (Elastic Net). The L1 regularization will shrink some parameters to zero. i.e. the coefficients of the variables that add minor value to the model will be zero. It adds a penalty equal to the sum of the absolute value of the coefficients.

The L2 regularization adds a penalty equal to the sum of the squared value of the coefficients. The L2 regularization will force the parameters to be relatively small. It guides the parameters to be close to zero, but not zero. It adds penalty equivalent to square of the magnitude of coefficients.

Ridge regression minimizes the sum squared residuals plus a penalty (lambda) on the number and size of the coefficients.

Alpha and lambda are the regularization strengths and must be a positive float. They act in a similar manner.

The bigger the penalization, the smaller (and the more robust) the coefficients are. Elastic-net is a mix of both L1 and L2 regularizations. A penalty is applied to the sum of the absolute values and to the sum of the squared values. Enhance Data Science. (2017). *Machine Learning Explained: Regularization*. Retrieved from: https://enhancedatascience.com/2017/07/04/machine-learning-explained-regularization/

In this paper, we perform various regularization exercises in R and report on the findings.

Analysis

We utilize the Boston Housing data set in R. It contains the housing data for 506 census tracts of Boston from the 1970 census. It contains 506 rows/ observation and 14 features/ variables.

```
> str(BostonHousing)
'data.frame':	506 obs. of  14 variables:
 $ crim   : num  0.00632 0.02731 0.02729 0.03237 0.06905 ...
 $ zn     : num  18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...
 $ indus  : num  2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...
 $ chas   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ nox    : num  0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524
...
 $ rm     : num  6.58 6.42 7.18 7 7.15 ...
 $ age    : num  65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...
 $ dis    : num  4.09 4.97 4.97 6.06 6.06 ...
 $ rad    : num  1 2 2 3 3 3 5 5 5 5 ...
 $ tax    : num  296 242 242 222 222 222 311 311 311 311 ...
 $ ptratio: num  15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...
 $ b      : num  397 397 393 395 397 ...
 $ lstat  : num  4.98 9.14 4.03 2.94 5.33 ...
 $ medv   : num  24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
```

Variables are as follows:

CRIM per capita crime rate by town

ZN proportion of residential land zoned for lots over 25,000 sq.ft.

INDUS proportion of non-retail business acres per town

CHAS Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)

NOX nitric oxides concentration (parts per 10 million)

RM average number of rooms per dwelling

AGE proportion of owner-occupied units built prior to 1940

DIS weighted distances to five Boston employment centres

RAD index of accessibility to radial highways

TAX full-value property-tax rate per $10,000

PTRATIO pupil-teacher ratio by town

B= 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town

LSTAT % lower status of the population

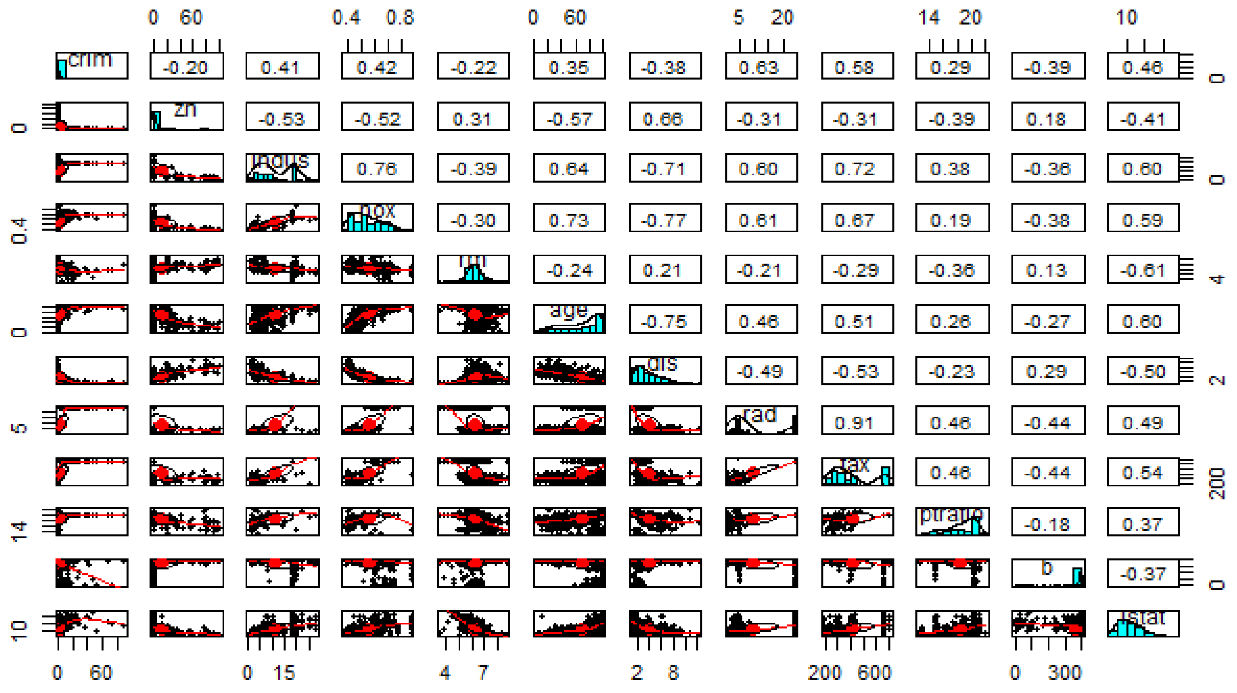
MEDV Median value of owner-occupied homes in $1000's

We set MEDV as our target variable.

First, we visually inspect the relationships between the independent variables and the correlation coefficients between the variables to test the strength of association between them. We exclude the target variable “MEDV” and the factor variable “chas”.

We achieve these two on the same plot using the pairs.panel() function in “psych” package in R as follows:

```
> library(psych)
> pairs.panels(Boston[c(-4,-14)])
```



As we can observe, some independent variables such as “tax” and “rad” are highly correlated leading to a multicollinearity problem which often time leads to overfitting thus a reducing the generalizations abilities of our predictive model. Multicollinearity is an incident where one or more of the independent variables are strongly correlated with each other. In such incidents, we should use only one among correlated independent variables.

We then model a linear regression model between our response and explanatory variables and use summary() function in R to get the summary statistics for our model on which we can make a model comparison.

Our objective is to select the model based on following characteristics:

- A fewer number of predictors is preferable
- Penalize a model having a lot of predictors
- Penalize a model for a bad fit

First, we partition our data to training and test data in a 70/30 split.

```
> library(caTools)
> #Set seed to ensure consistency of sample if picked again
> set.seed(123)
> split <- sample.split(Boston,SplitRatio =0.7)
> train <- subset(Boston,split==TRUE)
> test <- subset(Boston,split==FALSE)
```

Then we define a 10-fold cross-validation as a resampling technique, where the training data is divided into 10 parts. Each time the algorithm uses 9 parts to train the model and the 10th part to test the error and this is repeated among all the parts (recursive partitioning).

```
> Custom<-trainControl(method="repeatedcv",number=10,  repeats=5)
```

We then plot our linear model on the training data as follows:

```
> lm_model<- train(medv ~. , data = train, method='lm', trControl=Custom)
> summary(lm_model)

Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
     Min       1Q   Median       3Q      Max 
-16.3609  -2.3225  -0.3985   1.5479  24.2523 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  31.701646   6.181450   5.129 5.15e-07 ***
crim         -0.102177   0.032919  -3.104  0.00209 ** 
zn            0.041989   0.015998   2.625  0.00910 ** 
indus        -0.015372   0.067418  -0.228  0.81979    
chas1         3.845123   0.932691   4.123 4.81e-05 ***
nox         -18.038279   4.292505  -4.202 3.46e-05 ***
rm            4.240680   0.531755   7.975 2.94e-14 ***
age           0.003348   0.016342   0.205  0.83778    
dis          -1.417148   0.228738  -6.196 1.84e-09 ***
rad           0.231461   0.073040   3.169  0.00168 ** 
tax          -0.009996   0.004031  -2.480  0.01367 *  
ptratio      -0.891616   0.148181  -6.017 4.99e-09 ***
b             0.008631   0.003136   2.752  0.00627 ** 
lstat        -0.465039   0.066726  -6.969 1.91e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.312 on 311 degrees of freedom
```

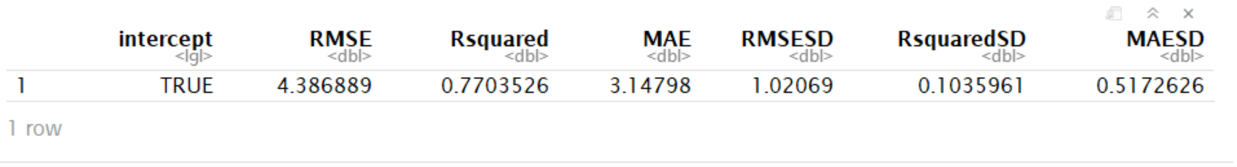
```
Multiple R-squared:  0.7822,   Adjusted R-squared:  0.7731
F-statistic: 85.91 on 13 and 311 DF,  p-value: < 2.2e-16
```

Our model returns a correlation coefficient (Multiple R-squared) of 0.7822 which is a fairly strong indicator of a positive linear relationship in the model.

The adjusted R-squared measures the variation of the dependent variables around the mean that are explained by the x-values. It tells how many points fall in the regression line. In our case, 77.31 % of variability in the dependent variable is explained by the independent variables.

To obtain further summary statistics, we run the code;

```
> lm_model$results
```



	intercept <lgl>	RMSE <dbl>	Rsquared <dbl>	MAE <dbl>	RMSESD <dbl>	RsquaredSD <dbl>	MAESD <dbl>
1	TRUE	4.386889	0.7703526	3.14798	1.02069	0.1035961	0.5172626

1 row

The RMSE indicates how close the predicted values are to the actual values; hence a lower RMSE value signifies that the model performance is good.

However, from our summary output, certain predictors such “indus” which returns a high p-value shows that it is not statistically significant.

We’ll leverage upon the regularization model to apply penalties to our coefficients in order that we can have a more statistically significant prediction model.

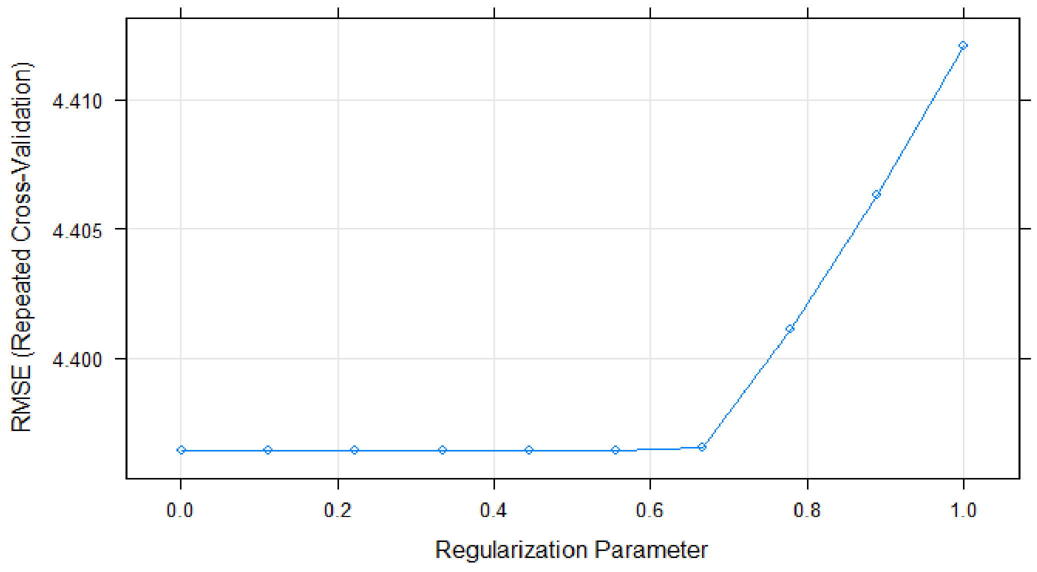
First, we model ridge regression on our data where all predictors are maintained but a penalty is applied to the magnitude of the predictors.

```
> #Ridge regression
> ridge<-train(medv~., data=train, method="glmnet", tuneGrid=expand.grid(alph
a=0,lambda=seq(0.001,1,length=10)), trControl=Custom)
> ridge
glmnet

325 samples
 13 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 5 times)
Summary of sample sizes: 291, 292, 293, 292, 293, 293, ...
Resampling results across tuning parameters:

  lambda  RMSE      Rsquared   MAE
  0.001   4.396449  0.7734052  3.080513
  0.112   4.396449  0.7734052  3.080513
  0.223   4.396449  0.7734052  3.080513
  0.334   4.396449  0.7734052  3.080513
  0.445   4.396449  0.7734052  3.080513
  0.556   4.396449  0.7734052  3.080513
  0.667   4.396520  0.7734040  3.080551
  0.778   4.401116  0.7731894  3.080674
  0.889   4.406297  0.7729319  3.081677
  1.000   4.412064  0.7726373  3.083729

Tuning parameter 'alpha' was held constant at a value of 0
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were alpha = 0 and lambda = 0.556.
```

The algorithm finds that the best value for lambda that will apply sufficient penalty to our coefficients to be 0.556.

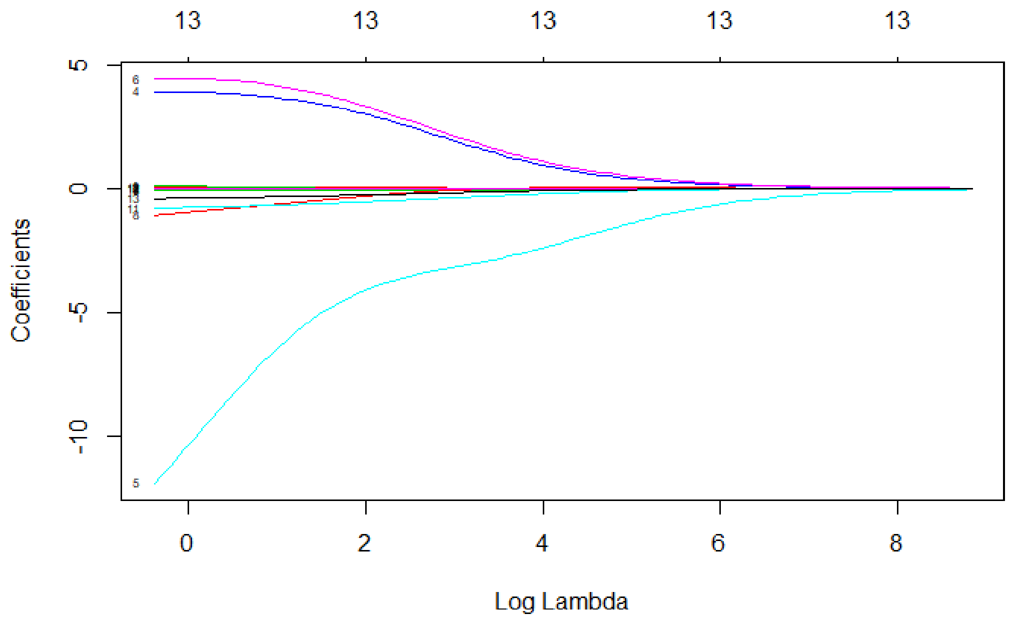
Plotting the ridge regression gives:

```
> plot(ridge)
> plot(ridge$finalModel, xvar="lambda",label=T)
```

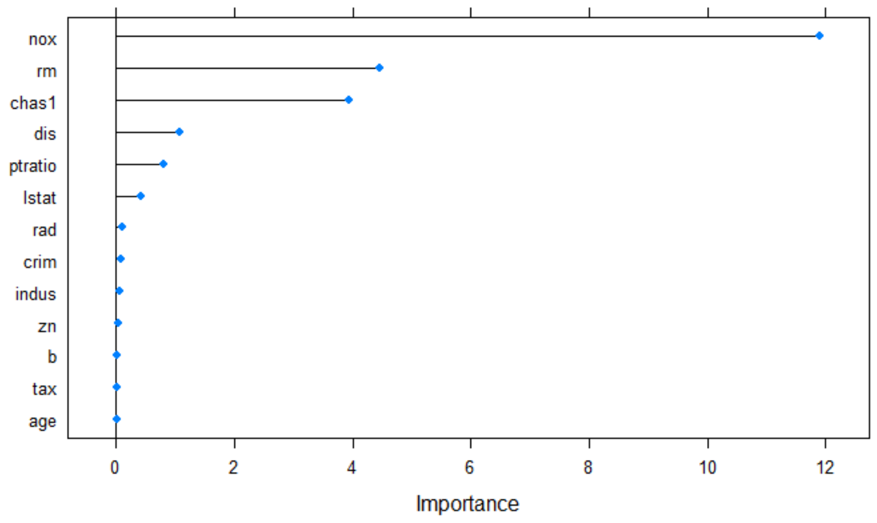


As see, on higher values of lambda, the test error increases meteorically.

In the plot below, we can observe that as log lambda increases the coefficients shrink towards zero.



Next, we plot the variable importance chart in R to know the scale of importance of variables in our ridge regression model.

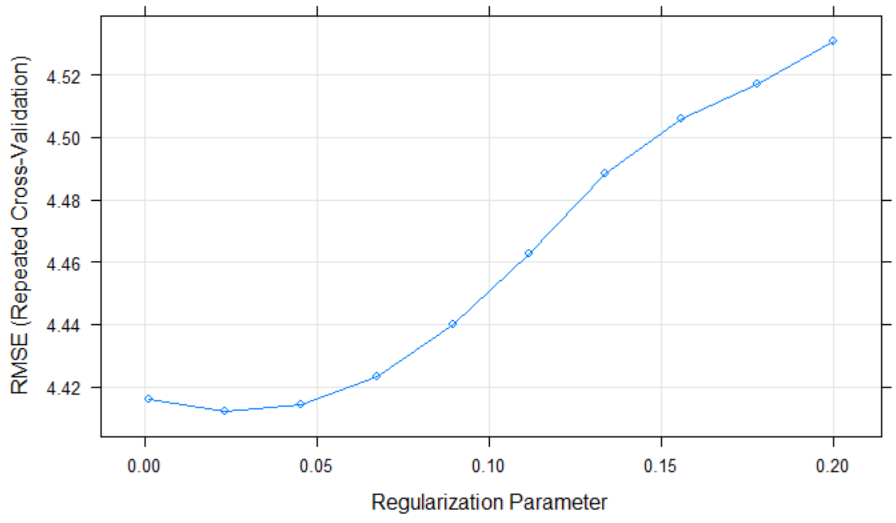


As seen, “nox” i.e. nitric oxides concentration variable is the the most important and should be retained in our model. “Age” is the least important variable.

Next, we plot a LASSO regression model which shrinks some parameters to zero and also ignores the multicollinear predictor variables.

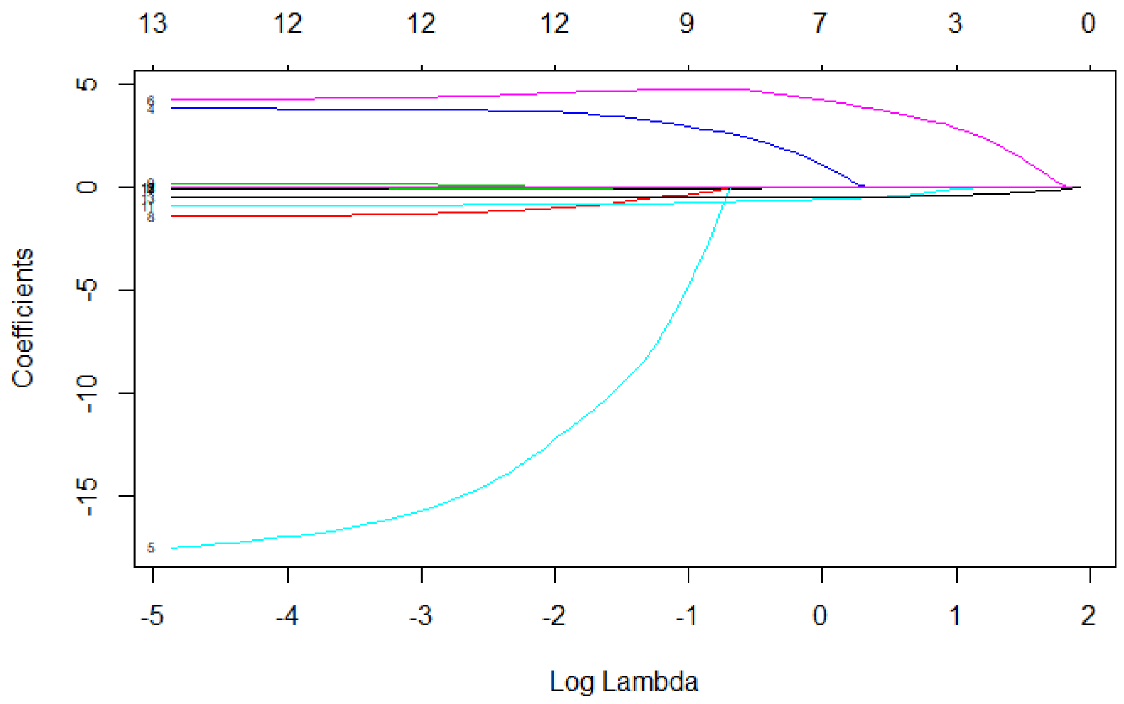
```
> #LASSO Regression
> set.seed(123)
> Lasso<-train(medv~., data=train, method="glmnet", tuneGrid=expand.grid(alph
a=1,lambda=seq(0.001,0.2,length=10)), trControl=Custom)
> Lasso
glmnet

325 samples
 13 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 5 times)
Summary of sample sizes: 293, 291, 293, 293, 293, 293, ...
Resampling results across tuning parameters:

  lambda      RMSE      Rsquared   MAE
  0.00100000  4.415737  0.7692816  3.131583
  0.02311111  4.412287  0.7696457  3.117910
  0.04522222  4.414138  0.7695489  3.105680
  0.06733333  4.423206  0.7687865  3.100536
  0.08944444  4.439787  0.7672702  3.101640
  0.11155556  4.462665  0.7650858  3.107891
  0.13366667  4.488364  0.7625394  3.117881
  0.15577778  4.505844  0.7608434  3.124044
  0.17788889  4.517049  0.7598249  3.127837
  0.20000000  4.530639  0.7586006  3.133871

Tuning parameter 'alpha' was held constant at a value of 1
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were alpha = 1 and lambda = 0.02311111.
```

This gives an optimal lambda value of 0.02311111.



Higher values of lambda increase the test error as observed above.

In the plot below, we can observe that as log lambda increases the coefficients shrink towards zero. Coefficients 6 and 4 perform better overall in their shrinkage and this means that they are influential in our model.



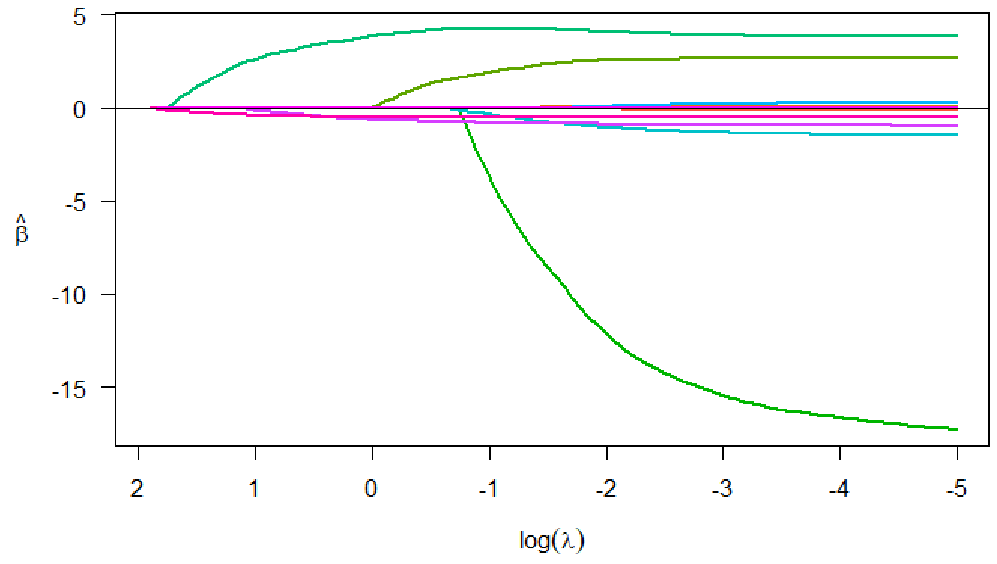
From the above plots, we can clearly see that when lambda is very small, the LASSO solution is very close to the OLS solution, and all of our coefficients are in the model. As lambda grows, the regularization term has greater effect and we see fewer variables in your model (because more and more coefficients will be zero valued).

Next, we extend big lasso on our dataset albeit it’s not such a big dataset.

Firs, we subset our data frame and convert our dependent and independent variables into a big matrix object as follows:

```
> X<-Boston[1:13]#Subsetting our data
> Y<-Boston[14]
> X<-as.big.matrix(X)
> Y<-as.big.matrix(Y)
> model_fit<-biglasso(X,Y, screen="SSR-BEDPP")# fit big lasso rule using scre
ening rule SSR- BEDPP
> plot(model_fit)
```

The same case as the LASSO regularization, when log lambda is small, all coefficients are in our model. However, as we increase the log lambda, some coefficients are shrunk to zero and therefore removed from our model.



We apply the summary() function to extract the optimal value for our regularization parameter lambda.

```
> model_fit <- cv.biglasso(X,Boston$medv, seed = 1234, nfolds = 10, ncores =
4)#Apply a 10-fold cross-validation
> summary(model_fit)
lasso-penalized linear regression with n=506, p=13
At minimum cross-validation error (lambda=0.0293):
-------------------------------------------------
  Nonzero coefficients: 11
  Cross-validation error (deviance): 23.74
  R-squared: 0.72
  Signal-to-noise ratio: 2.56
  Scale estimate (sigma): 4.872
```

As we can observe from our output, the optimal lambda value is 0.0293. We also note that the number of predictors in our model has reduced from 13 to 11.

Conclusion

In conclusion, any modelling technique is prone to overfitting in many cases when too many variables are included in the regression equation. The model may end up with spurious predictions. For most statistical techniques, the overfitting problem may be combatted by a judicious selection of predictor variables. Even with the application of regularization techniques, a judicious approach will be needed to ensure that the right variables are include in our model even those whose coefficients are shrunk to zero.

References

1. Peter B. (2017). *Practical Statistics for Data Scientists*. Boston. O Reilly Publishers.

2. Yaohui Zeng. (2016). *biglasso: extending lasso model to Big Data in R.* Retrieved from: https://cran.r-project.org/web/packages/biglasso/vignettes/biglasso.pdf