Regression analysis in R, just look at the Boston housing data and we can see a total of 506 observations and 14 variables.

In this dataset, medv is the response variable, and the remaining are the predictors.

We want to make a regression prediction model for medv based on other predictor variables.

Most of the variables are numeric variables except one variable.

First, we need to look at the multicollinearity problem, for that exclude factor variable.

In this case, some of the pairs are highly correlated and this may lead to inaccurate results.

**How to avoid collinearity issues?**

Collinearity leads to overfitting

The first solution is to fit ridge regression, shrinks coefficient to non-zero values to prevent overfitting, but keeps all variables.

The second option is lasso regression, which shrinks regression coefficients, with some shrunk to zero. Thus, it also helps with feature selection.

The third option is too elastic net regression, Mix of the ridge and lasso.

Elastic net regression sum of squares reduces to the ridge when alpha equals zero and reduces to lasso regression when alpha equals 1.

Elastic net regression models are more flexible. When we fit the elastic net regression model end up with the best model maybe 20% ridge and 80% lasso or it could be another combination of ridge and lasso.

**Regression analysis in R**

**Load Library**

library(caret)

library(glmnet)

library(mlbench)

library(psych)

**Getting Data**

data("BostonHousing")

data <- BostonHousing

**Data Partition**

set.seed(222)

ind <- sample(2, nrow(data), replace = T, prob = c(0.7, 0.3))

train <- data[ind==1,]

test <- data[ind==2,]

Custom Control Parameters with 10 number cross validation

custom <- trainControl(method = "repeatedcv",number = 10,repeats = 5,verboseIter = T)

**Linear Model**

set.seed(1234)

lm <- train(medv~.,train,methods='lm', trControl=custom)

Linear Regression

353 samples

 13 predictor

No pre-processing

Resampling: Cross-Validated (10 fold, repeated 5 times)

Summary of sample sizes: 316, 318, 318, 319, 317, 318, ...

Resampling results:

  RMSE     Rsquared  MAE

  4.23222  0.778488  3.032342

Tuning parameter 'intercept' was held constant at a value

 of TRUE

Call:

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

Residuals:

     Min       1Q   Median       3Q      Max

-10.1018  -2.3528  -0.7279   1.7047  27.7868

You can see RMSE is 4.23 and R squares is 0.77. Cross-validation is 10 indicates 9 parts used for training the model and one part used for testing the error and its repeated with five number of times.

summary(lm)

Coefficients:

              Estimate Std. Error t value Pr(>|t|)

(Intercept)  25.742808   5.653389   4.554 7.37e-06 \*\*\*

crim         -0.165452   0.036018  -4.594 6.15e-06 \*\*\*

zn            0.047202   0.015401   3.065 0.002352 \*\*

indus         0.013377   0.067401   0.198 0.842796

chas1         1.364633   0.947288   1.441 0.150630

nox         -13.065313   4.018576  -3.251 0.001264 \*\*

rm            5.072891   0.468889  10.819  < 2e-16 \*\*\*

age          -0.028573   0.013946  -2.049 0.041247 \*

dis          -1.421107   0.208908  -6.803 4.66e-11 \*\*\*

rad           0.260863   0.070092   3.722 0.000232 \*\*\*

tax          -0.013556   0.004055  -3.343 0.000922 \*\*\*

ptratio      -0.906744   0.139687  -6.491 3.03e-10 \*\*\*

b             0.008912   0.002986   2.985 0.003040 \*\*

lstat        -0.335149   0.056920  -5.888 9.40e-09 \*\*\*

---

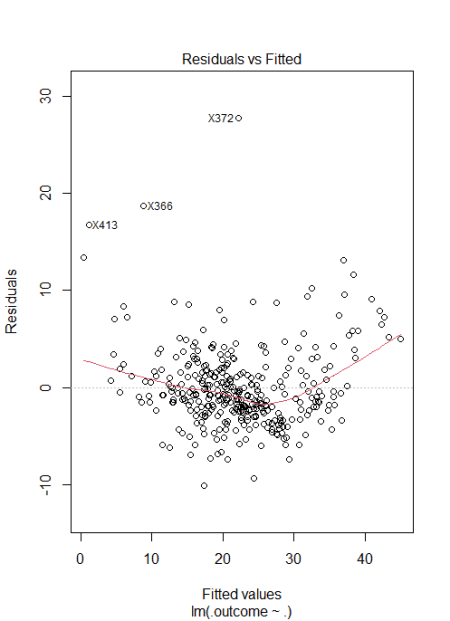
Signif. codes:  0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.192 on 339 degrees of freedom

Multiple R-squared:  0.7874,     Adjusted R-squared:  0.7793

F-statistic: 96.59 on 13 and 339 DF,  p-value: < 2.2e-16

The variables do not have a star indicates those variables are not statistically significant.

**Plot**

**Ridge Regression**

set.seed(1234)

ridge <- train(medv~.,train, method='glmnet',tuneGrid=expand.grid(alpha=0,lambda=seq(0.0001,1,length=5)),trControl=custom)

ridge

353 samples

 13 predictor

No pre-processing

Resampling: Cross-Validated (10 fold, repeated 5 times)

Summary of sample sizes: 316, 318, 318, 319, 317, 318, ...

Resampling results across tuning parameters:

  lambda    RMSE      Rsquared   MAE

  0.000100  4.242204  0.7782278  3.008339

  0.250075  4.242204  0.7782278  3.008339

  0.500050  4.242204  0.7782278  3.008339

  0.750025  4.248536  0.7779462  3.012397

  1.000000  4.265479  0.7770264  3.023091

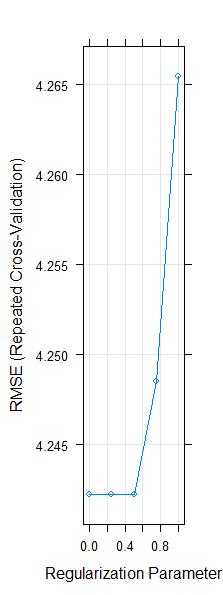
Tuning parameter ‘alpha’ was held constant at a value of 0

RMSE was used to select the optimal model using the smallest value.

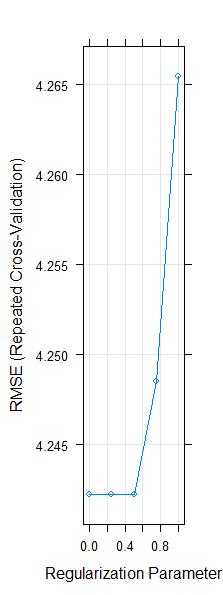
The final values used for the model were alpha = 0 and lambda = 0.50005.

You can see alpha is 0 because we are doing ridge regression and lambda is 0.5000.

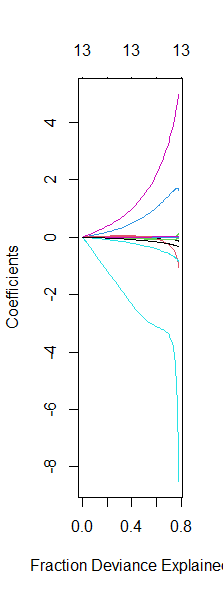
**Plot Results**

plot(ridge) 

Increase the lambda increases the error and the appropriate lambda is 0.5.

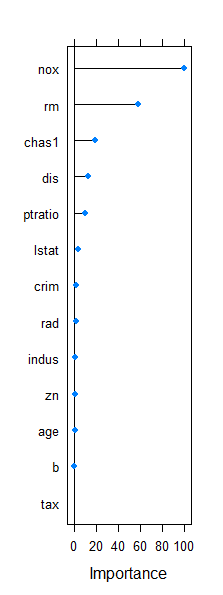
plot(ridge$finalModel, xvar = "lambda", label = T) 

X axis has log lambda, when log lambda around 9 all coefficients are zero.

plot(ridge$finalModel, xvar = 'dev', label=T) 

In this plot, you can see that the fraction deviation 60% model explains very well and after that lot of deviation noticed.

plot(varImp(ridge, scale=T))



The most important variables you can see in the top of the graph and at least once are at the bottom.

**Lasso Regression**

set.seed(1234)

lasso <- train(medv~.,train,

               method='glmnet',

               tuneGrid=expand.grid(alpha=1,

                                    lambda=seq(0.0001,1,length=5)),trControl=custom)

glmnet

353 samples

 13 predictor

No pre-processing

Resampling: Cross-Validated (10 fold, repeated 5 times)

Summary of sample sizes: 316, 318, 318, 319, 317, 318, ...

Resampling results across tuning parameters:

  lambda    RMSE      Rsquared   MAE

  0.000100  4.230700  0.7785841  3.025998

  0.250075  4.447615  0.7579974  3.135095

  0.500050  4.611916  0.7438984  3.285522

  0.750025  4.688806  0.7406668  3.362630

  1.000000  4.786658  0.7366188  3.445216

Tuning parameter ‘alpha’ was held constant at a value of 1

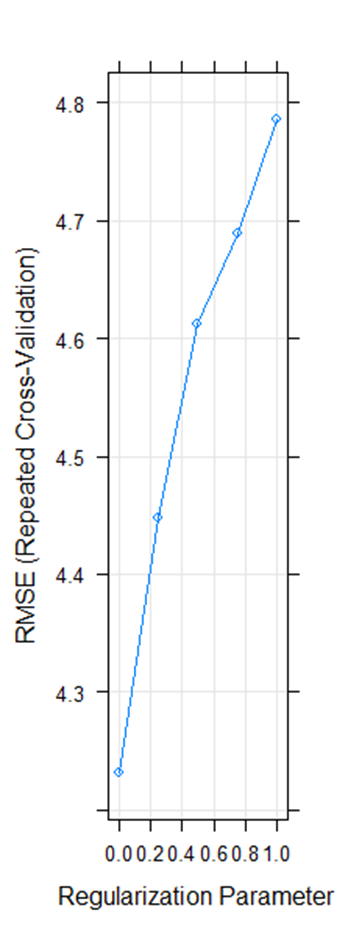
RMSE was used to select the optimal model using the  smallest value.

The final values used for the model were alpha = 1 and  lambda = 1e-04.

In this case lambda is close to zero that is the best value.

**Plot Results**

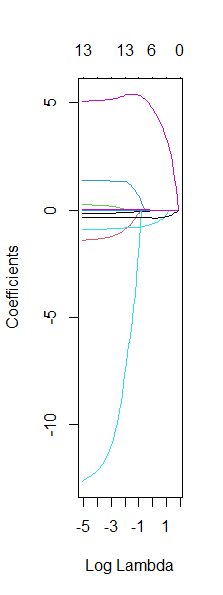
plot(lasso)

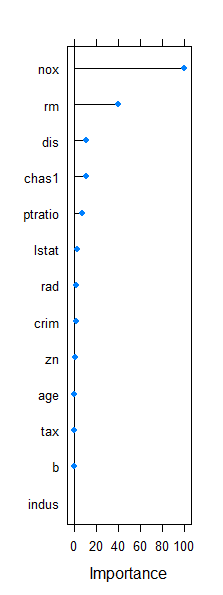


plot(lasso$finalModel, xvar = 'lambda', label=T)

60% of variability explains based on only 3 variables.

plot(varImp(ridge, scale=T))





Just look at the important 3 variables in lasso regression.

**Elastic Net Regression**

set.seed(1234)

en <- train(medv~.,train,

            method='glmnet',

            tuneGrid=expand.grid(alpha=seq(0,1,length=10),

                                 lambda=seq(0.0001,1,length=5)),trControl=custom)

glmnet

353 samples

 13 predictor

No pre-processing

Resampling: Cross-Validated (10 fold, repeated 5 times)

Summary of sample sizes: 316, 318, 318, 319, 317, 318, ...

Resampling results across tuning parameters:

  alpha      lambda    RMSE      Rsquared   MAE

  0.0000000  0.000100  4.242204  0.7782278  3.008339

  0.0000000  0.250075  4.242204  0.7782278  3.008339

  0.0000000  0.500050  4.242204  0.7782278  3.008339

  0.0000000  0.750025  4.248536  0.7779462  3.012397

  0.0000000  1.000000  4.265479  0.7770264  3.023091

0.1111111  0.000100  4.230292  0.7786226  3.025857

  0.1111111  0.250075  4.239094  0.7778348  3.005382

  0.1111111  0.500050  4.272822  0.7751270  3.024999

  0.1111111  0.750025  4.314170  0.7719071  3.052562

  0.1111111  1.000000  4.357845  0.7686150  3.085807

  0.2222222  0.000100  4.230694  0.7785669  3.026161

  0.2222222  0.250075  4.258991  0.7758849  3.015914

  0.2222222  0.500050  4.330452  0.7695318  3.059968

  0.2222222  0.750025  4.389640  0.7650387  3.106606

  0.2222222  1.000000  4.443160  0.7613804  3.151750

  0.3333333  0.000100  4.230795  0.7785677  3.026282

  0.3333333  0.250075  4.285269  0.7732992  3.030452

  0.3333333  0.500050  4.382444  0.7647643  3.096016

  0.3333333  0.750025  4.457291  0.7590837  3.157815

  0.3333333  1.000000  4.537080  0.7528068  3.229560

  0.4444444  0.000100  4.230574  0.7785789  3.025987

  0.4444444  0.250075  4.318752  0.7699550  3.049478

  0.4444444  0.500050  4.426926  0.7608447  3.127902

  0.4444444  0.750025  4.528733  0.7524128  3.216182

  0.4444444  1.000000  4.610942  0.7461712  3.292246

  0.5555556  0.000100  4.230656  0.7785681  3.026115

  0.5555556  0.250075  4.353828  0.7665028  3.071586

  0.5555556  0.500050  4.474680  0.7564421  3.164763

  0.5555556  0.750025  4.591765  0.7464771  3.269433

  0.5555556  1.000000  4.638309  0.7448745  3.323076

  0.6666667  0.000100  4.230688  0.7785626  3.026161

  0.6666667  0.250075  4.378865  0.7642222  3.087591

  0.6666667  0.500050  4.522902  0.7518766  3.203910

  0.6666667  0.750025  4.616421  0.7448532  3.295564

  0.6666667  1.000000  4.668353  0.7434801  3.351792

  0.7777778  0.000100  4.230768  0.7785606  3.026086

  0.7777778  0.250075  4.400658  0.7622860  3.101157

  0.7777778  0.500050  4.568780  0.7474490  3.243044

  0.7777778  0.750025  4.636481  0.7438164  3.317472

  0.7777778  1.000000  4.705950  0.7413472  3.383504

0.8888889  0.000100  4.230862  0.7785562  3.026279

  0.8888889  0.250075  4.423849  0.7601929  3.117267

  0.8888889  0.500050  4.599200  0.7446729  3.270369

  0.8888889  0.750025  4.660298  0.7424824  3.338783

  0.8888889  1.000000  4.746398  0.7389209  3.415104

  1.0000000  0.000100  4.230700  0.7785841  3.025998

  1.0000000  0.250075  4.447615  0.7579974  3.135095

  1.0000000  0.500050  4.611916  0.7438984  3.285522

  1.0000000  0.750025  4.688806  0.7406668  3.362630

  1.0000000  1.000000  4.786658  0.7366188  3.445216

RMSE was used to select the optimal model using the

 smallest value.

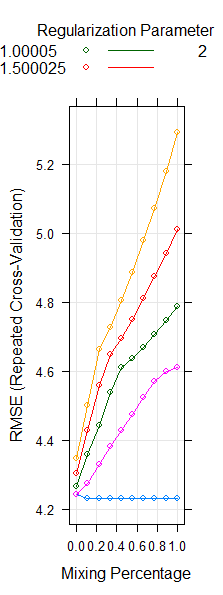
The final values used for the model were alpha = 0.1111111

 and lambda = 1e-04.

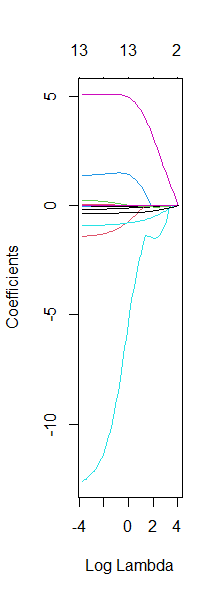
Now you can see that alpha= 0.111 and lambda=1e-04.

**Plot Results**

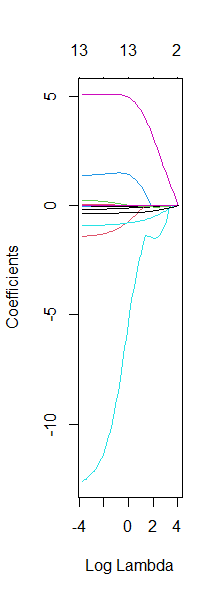
plot(en)



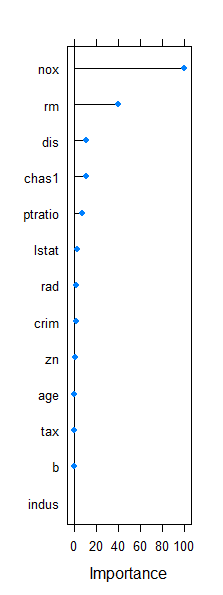
plot(en$finalModel, xvar = 'lambda', label=T)



plot(en$finalModel, xvar = 'dev', label=T)



plot(varImp(en))



**Compare Models**

Now just compare the models we created,

model\_list <- list(LinearModel=lm,Ridge=ridge,Lasso=lasso,ElasticNet=en)

res <- resamples(model\_list)

summary(res)

Call:

summary.resamples(object = res)

Models: LinearModel, Ridge, Lasso, ElasticNet

Number of resamples: 50

MAE

                Min.  1st Qu.   Median     Mean  3rd Qu.

LinearModel 2.080208 2.767061 3.002455 3.032342 3.355281

Ridge       2.094151 2.736246 2.934350 3.008339 3.366834

Lasso       2.072408 2.764289 2.988132 3.025998 3.346437

ElasticNet  2.074008 2.762076 2.987955 3.025857 3.348605

                Max. NA's

LinearModel 3.874270    0

Ridge       3.971337    0

Lasso       3.882800    0

ElasticNet  3.882943    0

RMSE

                Min.  1st Qu.   Median     Mean  3rd Qu.

LinearModel 2.673817 3.495197 3.998562 4.232220 4.751509

Ridge       2.478993 3.477912 4.169422 4.242204 4.759265

Lasso       2.650331 3.490881 3.993362 4.230700 4.748958

ElasticNet  2.650603 3.489053 3.993227 4.230292 4.747517

                Max. NA's

LinearModel 7.027551    0

Ridge       7.035089    0

Lasso       7.040494    0

ElasticNet  7.033125    0

Rsquared

                 Min.   1st Qu.    Median      Mean   3rd Qu.

LinearModel 0.4865769 0.7269864 0.7991104 0.7784880 0.8472274

Ridge       0.4796929 0.7339342 0.8018589 0.7782278 0.8459744

Lasso       0.4848588 0.7272700 0.8002386 0.7785841 0.8475939

ElasticNet  0.4855896 0.7271484 0.8002849 0.7786226 0.8476337

                 Max. NA's

LinearModel 0.9128278    0

Ridge       0.9141020    0

Lasso       0.9138499    0

ElasticNet  0.9134723    0

Elastic Net regression model comes as best fit model based on RMSE.

**Best Model**

en$bestTune

best <- en$finalModel

coef(best, s = en$bestTune$lambda)

You can find out best coefficients based on above command.

**Prediction**

p1 <- predict(fm, train)

sqrt(mean((train$medv-p1)^2))

4.108671

p2 <- predict(fm, test)

sqrt(mean((test$medv-p2)^2))

6.14675

**Conclusion**

If we look at the RMSE the lowest value coming in the elastic net model. Elastic Net regression model avoids multicollinearity issue and provides the best model.