R Assignment 1 - Lasso Regression

Jasmeet Singh Saini

2023-01-26

Introduction

One of the most helpful techniques in a statistician's toolkit to overcome **overfitting** of model is *Lasso regression*. *Lasso regression* uses a technique that is frequently used to increase the flexibility of your model is *Regularisation*.

Regularization is one of the methods to improve the adaptability of the model. It is a crucial idea that helps prevent *overfitting* of the data, especially when the trained and test sets of data differ significantly explained in **Figure 1**.

In this techniques, the coefficient estimates are constrained, regularised, or shrunk in the direction of zero. In other words, to reduce the chance of *overfitting*, this strategy opposes learning a more sophisticated or flexible model. For a more accurate forecast, it is preferred over regression techniques.

Below is the straightforward linear regression relationship:-

$$Y \approx \beta 0 + \beta 1X1 + \beta 2X2 + ... + \beta pXp$$

where,

Y = represents the learned relation,

β = represents the coefficient estimates for different variables or predictors(X).

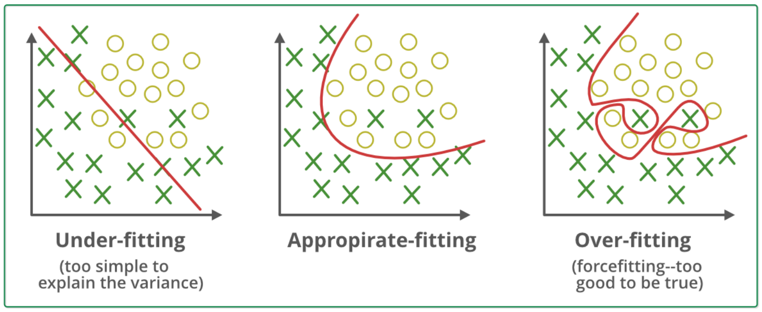
Under-fitting
(too simple to explain the variance)
Appropirate-fitting
Over-fitting
(forcefitting--too good to be true)

Figure 1: The problem faced in Regularization are Under-fitting, Appropirate-fitting, Over-fitting

The **residual sum of squares** (also known as RSS), loss function is used during the fitting process. This loss function's minimization is achieved by selecting the coefficients.

$$RSS = \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2$$

RSS will now change the coefficients in accordance with the training data. The estimated coefficients won't generalise well to the subsequent data if there is noise in the training data. At this point, *Regularization* shrinks or regularises these learnt estimates in the direction of zero.

Description

The "absolute value of magnitude" of the coefficient is added as a penalty term by the Lasso Regression to the loss function (L). Lasso regression is also known as *L1 regularization* or *L1 norm*

This form of regularization can produce sparse models with few coefficients; certain coefficients may become zero and so be removed from the model. Larger penalties provide coefficient values that are closer to zero, which is great for creating simpler models. L2 regularisation, on the other hand (e.g., Ridge regression), does not result in the deletion of coefficients or sparse models. As a result, the Lasso is significantly easier to understand than the Ridge.

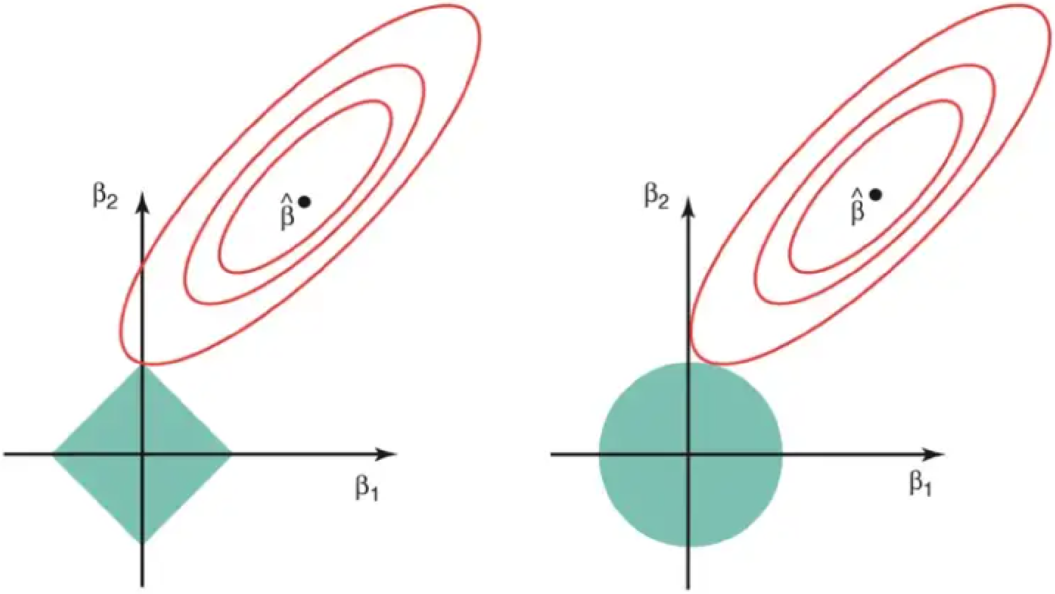

Figure 2: Depicts the constraint functions (green regions) for lasso (left side) and ridge regression (right side), as well as RSS contours (red ellipse).

This means that for all locations within the diamond given by $|\beta 1| + |\beta 2| \leq s$, lasso coefficients have the smallest RSS value(loss function).

$$|\beta 1| + |\beta 2| \leq s$$

Here, s is a *constant* that exists for all shrinkage factor λ. Constrained functions is the another name.

The RSS value is shared by all points on the ellipse. For very large values of s, the green patches will contain the ellipse's center, resulting in coefficient estimates from both regression techniques being equivalent to least squares estimates. However, this is not the case in the preceding photograph.

The first point at which an ellipse contacts the constraint zone gives the lasso and ridge regression coefficient estimations in this situation. Ridge regression has a circular restriction with no sharp points, this intersection will not usually occur on an axis, and hence the ridge regression coefficient estimations will be entirely non-zero. The lasso constraint, on the other hand, contains corners at each of the axes, therefore the ellipse will frequently meet the constraint region at an axis.

When this happens, one of the coefficients equals zero. In higher dimensions (parameters greater than 2), many of the coefficient estimates may equal 0 at the same time, refer **Figure 2**

"**LASSO**" is a statistical formula for feature selection and regularization of data models:-

- **L**east,
- **A**bsolute,
- **S**hrinkage,
- **S**election,
- **O**perator

The equation of Lasso Regression is,

Residual Sum of Squares + λ * (Sum of the absolute value of the magnitude of coefficients).

Here,

Lasso = Loss + penalty

Residual Sum of Squares (RSS) is *Loss*,

λ * (Sum of the absolute value of the magnitude of coefficients) is *penalty*.

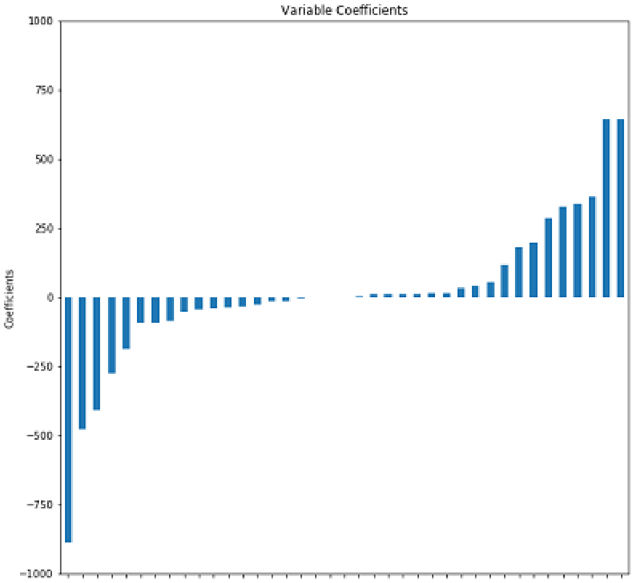

Figure 3: As we increase the value of λ, the magnitude of the coefficients decreases, where the values reaches to zero but not absolute zero.

Or, mathematical equation of Lasso Regression is as follow:-

$$= \sum_{i=1}^{n} \left(y_i - \beta_0 \sum_{i=1}^{n} \beta_i x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j$$

else,

$$= RSS + \lambda \sum_{j=1}^{p} \beta_j$$

where,

RSS = Residual Sum of Squares (as explained above),

β_j = Coefficient estimates,

λ = a hyper-parameter or amount of shrinkage Known as **regularization constant** and it is *greater than zero*. However, the following will happen:

- if bias increases then λ increases, $\propto \lambda$
- if variance increases then λ decrease, $\propto 1/\lambda$

When we have correlated variables, the fundamental issue with lasso regression is that it only keeps one variable and puts the other connected variables to zero, see **Figure 3**. That might cause some information to be lost, which would impair our model's accuracy.

Example

We will use the R built-in data-set (data-frame) called **mtcars**.

To explain *Lasso regression*, we will use **hp** as the *response variable* and the following variables as the *predictors*:

Number	Variables	Datatype	Description
1	hp	numeric	Gross horsepower
2	mpg	numeric	Miles/US Gallon
3	wt	numeric	Weight (lb/1000)
4	drat	numeric	Rear axle ratio
5	qsec	numeric	1/4 mile time

There are three basic steps, which will be performed:

1. Load the Data,

We will apply **glmnet** package, to perform *Lasso regression*.

```
library(glmnet)
horse_power <- mtcars$hp    #response variable
predictor_variable <- data.matrix(mtcars[, c('mpg', 'wt', 'drat', 'qsec')])
```

2. Fit the Lasso Regression Model,

In this step we will define the value of λ using glmnet() and get the lowest test mean squared error (MSE). cv.glmnet function will performs k-fold cross validation using k = 10 folds.

```
cv_model <- cv.glmnet(predictor_variable, horse_power, alpha = 1)

best_lambda <- cv_model$lambda.min
best_lambda
```

```
## [1] 2.431182
```

2.215202 turns out to be the λ value that minimizes the test MSE.

3. Analyze Final Model

Finally, we can examine the final model that the ideal λ value generated and find coefficient estimates. As a result, **drat** is empty because the lasso regression shrunk the coefficient all the way to *zero*.

```
plot(cv_model,col= rgb(0.5,0,0.5,alpha=0.6),xlab="Log",ylab="Mean-squared Error")
```

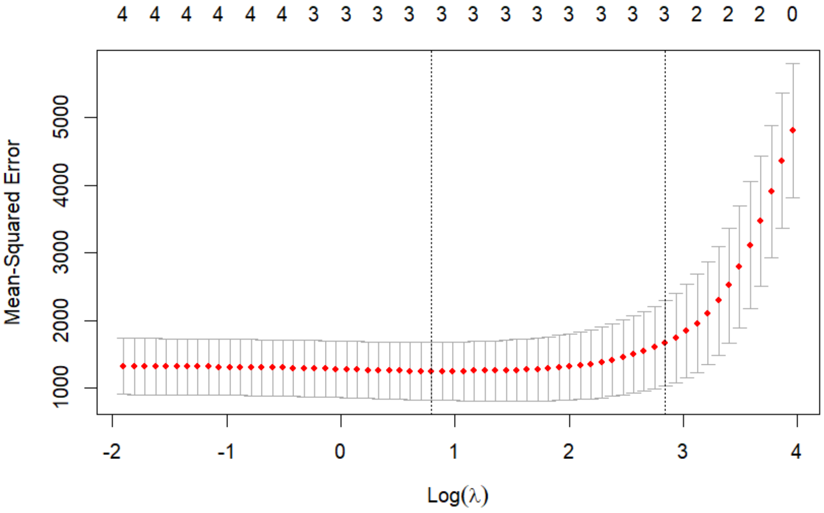

Figure 4: **Means Squared vs Lambda**

Now, we will make predictions on new observations in lasso regression. For example, let us suppose we have new car and its has the following attributes:

Variables	Value
mpg	24
wt	2.5
drat	3.5
qsec	18.5

The provided inputs gives the predicted value of hp as **109.0842**.

Now, its last step to calculate the R-squared on the training data, which comes out to be **0.8047064**, that is, **80.47%**.

Appendix - Code

```
horse_power <- mtcars$hp                #Response variable

#Here, a matrix of predictor variables
predictor_variable <- data.matrix(mtcars[, c('mpg', 'wt', 'drat', 'qsec')])

# perform k-fold cross-validation to find optimal lambda value
```

```
cv_model <- cv.glmnet(predictor_variable, horse_power, alpha = 1)
best_lambda <- cv_model$lambda.min
best_lambda             #find optimal lambda value that minimizes test MSE
```

```
## [1] 4.248559
```

```
plot(cv_model,col= rgb(0.5,0,0.5,alpha=0.6),xlab = "lambda",ylab = "Mean-squared Error")
```

```
#find coefficients of best model
best_model <- glmnet(predictor_variable, horse_power, alpha = 1, lambda = best_lambda)
# coef(best_model)
# drat shows Zero in coef(best_model) because lasso regression shrunk the coefficient.

new_obs = matrix(c(24, 2.5, 3.5, 18.5), nrow=1, ncol=4)    #New observation

#Predict response value, lasso regression model has been used
predict(best_model, s = best_lambda, newx = new_obs)
```

```
##            s1
## [1,] 108.6783
```

```
#use fitted best model to make predictions
predicted <- predict(best_model, s = best_lambda, newx = predictor_variable)
sst <- sum((horse_power - mean(horse_power))^2)           #find SST and SSE
sse <- sum((predicted - horse_power)^2)
rsq <- 1 - sse/sst                                       #find R-Squared
round(rsq,4)
```

```
## [1] 0.7996
```

Bibliography

- An **Introduction to Statistical Learning** by *Gareth James*, *Daniela Witten*, *Trevor Hastie*, *Robert Tibshirani.*(ISBN: 978-1-4614-7137-0, Series ISSN 1431-875X)
- **Regularization in machine learning**, https://www.statology.org/lasso-regression-in-r/
- **Datacamp: Lasso Regression**
- **Lasso Regression in R**, https://www.statology.org/lasso-regression-in-r/