A Comparative study on Linear, Ridge and Lasso Regression models for data containing multi-collinear variables

Parthipan V, Satish Vavilapalli

Abstract:

Regression analysis is one of the popular statistical method used in almost all industries and used to determine the strength of the relationship between one dependent variable (usually denoted by Y) and a series of other changing variables (known as independent variables). Many techniques are there for regression analysis and one of the familiar method is linear regression.

In Regression, many errors may occur such as Non linearity of the response and predictor relationships, correlation of error terms, Non constant variance of error terms, outliers, high leverage points and multicollinearity. Among these, Multicollinearity among the variables is one of the major problem in regression analysis. We can reduce that multicollinearity by using regularized regressions such as Ridge regression and Lasso regression.

This study is concentrating more about comparison on linear, ridge and lasso regression models for data containing multicollinear variables by taking a use case. The dataset is solved using all the three methods individually. The result will explain about the advantages and disadvantages of each of the models that exists among each other. It also explains about how these models handle the errors of the test data due to the presence of multicollinearity among the variables.

The conclusion can be drawn that this comparative study of these three models will result in which model handles multicollinearity in easier manner than the other two methods. All the required calculations and graphical displays are performed using the R software for statistical computing.

1 Introduction:

Multicollinearity (also collinearity) is a phenomenon in which one predictor variable in a multiple regression model can be linearly predicted from the others with a substantial degree of accuracy. In this situation the coefficient estimates of the multiple regression may change erratically in response to small changes in the model or the data. In this situation the coefficient estimates of the multiple regression may change erratically in response to small changes in the model or the data. Ridge and Lasso regressions are forms of regularized regression. Ridge regression will help us to automatically control the errors in the model while the Lasso Regression help us to automatically select the required attributes. These methods are seeking to reduce the consequences of multicollinearity. Regularization imposes an upper threshold on the values taken by the coefficients, thereby producing a more parsimonious solution, and a set of coefficients with smaller variance. Linear regression also can handle the effects of multicollinearity by using a concept of variance inflation factor (VIF) which detects the multicollinear attributes.

2 Multicollinearity:

3.1 What is Multicollinearity?

A key goal of regression analysis is to isolate the relationship between each independent variable and the dependent variable. The interpretation of a regression coefficient is that it represents the mean change in the dependent variable for each 1 unit change in an independent variable when you hold all of the other independent variables constant.

When independent variables are correlated, it indicates that changes in one variable are associated with shifts in another variable. The stronger the correlation, the more difficult it is to change one variable without changing another.

It becomes difficult for the model to estimate the relationship between each independent variable and the dependent variable independently because the independent variables tend to change simultaneously.

Multicollinearity causes the following two basic types of problems:

- The coefficient estimates can swing wildly based on which other independent variables are in the model. The coefficients become very sensitive to small changes in the model.
- Multicollinearity reduces the precision of the estimate coefficients, which weakens the statistical power of your regression model. You might not be able to trust the p-values to identify independent variables that are statistically significant.

2.2 Multicollinearity in Linear Model:

In matrix form, the linear model can be represented as :

$$Y = X\beta + \epsilon$$

By the least square method, we can get that solutions:

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

In the case of *perfect* multicollinearity (in which one independent variable is an exact linear combination of the others) the matrix X has less than full rank and also its determinant is zero and therefore the moment matrix $X^T X$ cannot be inverted. Under these circumstances, for a general linear model $= X\beta + \epsilon$, the ordinary least-squares estimator $\hat{\beta} = (X^T X)^{-1} X^T Y$ does not exist.

2.3 Detecting multicollinearity:

It can be detected using variance inflation factor (VIF). The VIF estimates how much the variance of a regression coefficient is inflated due to multicollinearity in the model.

VIFs are calculated by taking a predictor, and regressing it against every other predictor in the model. This gives you the R-squared values, which can then be plugged into the VIF formula. “i” is the predictor we are looking at (e.g. x1 or x2):

$$VIF = \frac{1}{1 - R_i^2}$$

A VIF of 1.9 tells you that the variance of a particular coefficient is 1.9 times bigger than what you would expect if there was no multicollinearity. If VIF is 1, the predictors are not correlated, between 1 to 5 moderately correlated and above 10, highly correlated.

3 Regression Models:

3.1 Linear regression:

Linear regression, a very simple approach of supervised learning, is a useful tool for predicting a quantitative response. In statistics, linear regression is a linear approach to modelling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). The case of one explanatory variable is called simple linear regression. For more than one explanatory variable, the process is called multiple linear regression.

Given a data set {Yi, Xi1, …., Xip} (for i=1 to n) of n statistical units, a linear regression model assumes that the relationship between the dependent variable Y and the p-vector of regressors X is linear. This relationship is modelled through a disturbance term or error variable ε — an unobserved random variable that adds "noise" to the linear relationship between the dependent variable and regressors. The matrix X has n rows and p columns, representing the p variables for n instances while the target vector y as a column vector of length n containing the corresponding values of the target variable. In general, the response y may be related to k regressors or predictor variables. Thus the multiple linear regression model with k regressors takes the form:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k + \varepsilon$$

The parameters β_j where j $= 0, 1, \ldots, k$ are called the regression coefficients. They represents the expected change in the response y per unit change in xi when all the remaining regressors xi ($i ! = j$) are held constant. We can use ordinary least square method for this parameter estimation which is usual way so in this section we are going to discuss result in matrix format equation. It is more convenient to deal with multiple regression models if they are expressed in matrix notation. This allows a very compact display of the model, data, and results. In matrix notation, the model given by:

$$Y = X\beta + \epsilon$$

By the least square method, we can get that solutions:

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

The estimates above are found using linear model using R program.

3.2 Ridge regression:

The most popular form of regularized regression is Ridge Regression. Ridge regression shrinks the regression coefficients by imposing a penalty on the sum of squares (L2 norm) of the regression coefficients. The use of an L2-penalty in least-squares problem is sometimes refer to as Tikhonov regularization (the method is also known as ridge regression).

In Linear regression, we choose the estimates by minimizing its cost function through optimization and the model with these estimates fits the data well. However, this doesn't control the overfitting of the model. So, apart from fitting only the model well with the data, we need to control the overfitting also. So, here to overcome this, Ridge regression plays a role with the modified cost function and help us to control fit of the model as well as controlling the magnitude of the coefficient estimates by controlling the overfitting of the model. The Ridge cost function is:

$$\hat{\beta}_{ridge} = \arg\min \left\{ \sum_{i=0}^{n} (Y_i - X_i\beta)^2 + \lambda \sum_{j=0}^{p} \beta_j^2 \right\}$$

The left part of the term shown above is the usual least squares criterion. In the right part, λ is a shrinkage factor applied to the sum of the squared values of the regression coefficients. The smaller value of λ will be leading to the state that all the coefficient estimates are same as the one from the OLS results. Also the larger the value of λ, the heavier the penalty on the regression coefficients, and the more they are shrunk towards zero. In ridge regression, all the variables stay in the model since regression coefficients do not become exactly zero (that would be equivalent to variables dropping out of the regression model). Ridge gives equal weight to absolutely correlated variables in the data set. The ridge estimators can be found as:

$$\hat{\beta}_{ridge} = (X^T X + \lambda I)^{-1} X^T y$$

3.3 Lasso Regression:

Lasso (LASSO - least absolute shrinkage and selection operator) is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces. We have seen that ridge regression essentially re-scales the OLS estimates. The lasso, by contrast, tries to produce a sparse solution, in the sense that several of the slope parameters will be set to zero. Here the penalty is applied to the sum of the absolute values of the regression coefficients, the L1 norm. The L1-penalty is solely applied to the slope coefficients, and thus the intercept $\beta 0$, is excluded from the penalty term. The Lasso function can be represented as:

$$\hat{\beta}_{lasso} = \arg\min \left\{ \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda \sum_{j=0}^{p} |\beta_j| \right\}$$

Again, the left part of the term is the normal least squares criterion. The right part now is the penalized sum of the absolute values of the regression coefficients. Similar to ridge regression, the shrinkage parameter (λ) has to be found. Penalizing the absolute values of the regression coefficients has the effect that a number of the estimated coefficients will become exactly zero, which means that some attributes drop out of the regression model so that a LASSO fitted model will consist of fewer variables than the original number of available attributes. In other words, LASSO can implicitly perform variable selection. The number of selected variables is upper limited by the numbers of samples (n). In case of absolutely correlated variables, LASSO just selects one of these variables and ignores the rest in the group.

3.4 Selecting the λ value for Ridge and LASSO (Cross-validation):

Cross-validation is the technique that can be used to find a "proper" value for the λ parameter given a sample. Here "proper" means finding a λ that would allow to predict the response values with the highest accuracy. It is clear that the λ values that are too small can lead to ovefitting when the model would tend to describe the noise in the data. Too large λ values, on the contrary, would lead to under fitting when the procedure cannot capture the underlying relationship. In both cases we will get a high error value when calculated on the test data (a set of observations not included into the initial sample). To perform cross-validation, we first divide the initial data into two subsets: one is called the train set and

the other one is called the test set. Then the train set is used to calculate the coefficient estimates. These estimates are then validated on the test set.

Let us now describe the algorithm in some more detail. First the initial data set is randomly divided into K blocks of equal length. One of the blocks is assigned the role of the test set while the remaining K – 1 blocks together constitute the train set. In practice the number of blocks K is usually selected to be 5 or 10. Next we choose a grid of values $\lambda = \lambda_s$ and calculate the regression coefficients for each λ_s value. Given these regression coefficients, we then compute the residual sum of squares:

$$RSS_{\lambda_s,k} = \sum_{i=1}^{n}\left(y_i - \sum_{j=0}^{k-1}\hat{\beta}(k,\lambda_s)x_{ij}\right)^2$$

Where k = 1,….K is the index of the block selected as the test set. One can obtain the average of these RSS values over all blocks.

$$MSE_{\lambda_s} = \frac{1}{K}\sum_{k=1}^{K} RSS_{\lambda_s,k}$$

λ is then set equal to s that gives the minimum MSE_{λ_s}.

Another popular approach utilizes the “one-standard-error rule”. For each MSE λ_s the standard error of the mean is calculated. Then we select the largest λ_s for which the MSE λ_s is within one standard error of the minimum MSE λ_s value. This way we obtain a “more regularized” model while increasing the MSE λ_s by not more than one standard error.

4 Applying the Models to data:

4.1 Dataset:

The dataset used in this study is Wine Quality dataset from UCI Machine Learning Repository. It has two datasets that are related to red and white variants of the Portuguese "Vinho Verde" wine. Here, we have considered only the red wine data. There are 12 predictors and around 1599 observations. Among these 12 predictors, 11 are predictors which are all physiochemical wine parameters and 1 target variable which is wine quality score. The score ranges from 0 to 10.

Assumption:

The target variable of wine score is ranging from 0 to 10 integer numbers which may be classification problem. So, the main assumption that we took here is considering this wine score as target which ranges from 0 to 10 as continuous numeric value having decimal values of score. Then this data is treated as regression problem.

4.2 Linear Regression, Ridge and LASSO Models applied:

Initially the dataset is fitted with the linear regression model using OLS in R using the package lm (). Then, Ridge and LASSO models are applied using “glmnet” library in R.

```
library(glmnet)
```

The wine dataset is in CSV format. And also, it did not have initially column names. The column names are provided separately. We have edited manually added the column names to the CSV file for easier reading of the dataset. The dataset is saved as data frame named df. The column names also given in table 1.

```
df = read.csv("winequality-red.csv")
dim(wine)
[1] 1599 12
colnames(df)
```

Table 1. Attribute names of the red wine dataset:

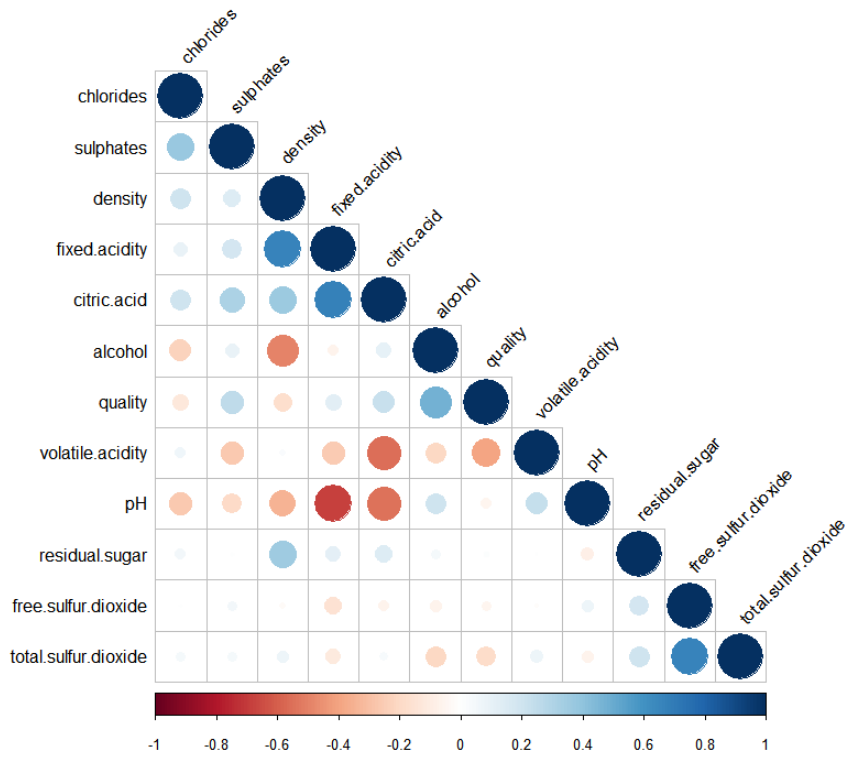
```
 [1] "fixed.acidity"        "volatile.acidity"     "citric.acid"
 [4] "residual.sugar"       "chlorides"            "free.sulfur.dioxide"
 [7] "total.sulfur.dioxide" "density"              "pH"
[10] "sulphates"            "alcohol"              "quality"
```

The data need to be checked for the correlation between each other via correlation matrix and it can be plotted also as correlation plot using `corrplot` library which can be found in Figure 1. It can be found from the figure 1 that other than the dependent variable "quality", there exist the high correlation among the predictors also stating the presence of multicollinearity in the predictors.

```
library(corrplot)

View(cor(df))
corrplot(cor(df), type = "lower", order = "hclust",
         tl.col = "black", tl.srt = 45)
```

Figure 1. Correlation Plot of variables:



To measure the quality of each regression models, we are going to validate the model in test data. So, we randomly divided the whole data into two subsets: train and test datasets. Here set.seed is set as fixed value of 1 so that the fixed results can be produced.

```
set.seed(1)

rand = sample(1:nrow(df),as.integer(dim(df)[1] * 0.5))
train = df[rand, ]
test = df[-rand, ]
```

To perform the linear regression model, lm command is used. The OLS coefficients can be found in table 3 (Note: The coefficients are mentioned in table 3 along with the coefficients of Ridge and LASSO). The summary command gives the overview of the different regression model parameters. One of them is the coefficient of determination R^2 that in our case is 0.3334006.

```
m1 <- lm(quality ~ ., train)
summary(m1)
summary(m1)$r.square
[1] 0.3334006
```

As we know that the dataset that we have fitted in linear regression contains multicollinearity, we have to detect those variables. It can be detected by the variance inflation factors (VIF) which is available in car library.

```
library(car)
vif(lm.mod)
```

Table 2. Variance inflation factors (VIF) for 11 predictors:

fixed.acidity	volatile.acidity	citric.acid	residual.sugar	chlorides	free.sulfur.dioxide	total.sulfur.dioxide	density	pH	sulphates	alcohol
7.175034	1.77755	3.424212	1.673862	1.513932	2.043192	2.235502	5.874435	3.239267	1.447768	2.85295

The VIF values are given in table 2. Some of these values are rather high, like “fixed.acidity” and “density”. This indicates the presence of multicollinearity.

Now, the dataset is applied to the Ridge and LASSO models. As it was already mentioned, different λ values will produce different coefficient estimates β. In this case λ takes 100 different values from 10^{-2} to 10^{5}. In R, we can visualize how the β estimates change depending on λ for a given sample. To do this, we first choose a range of λ values and build regression models for each λ value from this range using glmnet package.

```
lambda.grid = 10^seq(5, -2, length=100)
x <- model.matrix(quality ~ 0 + . , data=train)
```

The Ridge model can be applied by setting alpha value equals 0 while in the case of LASSO it is 1. The graphs for the respective models also can be plotted for different λ values agains the coefficient estimates β.

```
## Ridge model applying for different lambda values:
ridge.mod = glmnet(x, train$quality, alpha=0, lambda=lambda.grid)
plot(ridge.mod, xvar='lambda')
```

```
## LASSO model applying for different lambda values:
lasso.mod = glmnet(x, train$quality, alpha=1, lambda=lambda.grid)
plot(lasso.mod, xvar='lambda')
```

Figure 2. Coefficient estimates β for Ridge regression for the red wine data plotted versus log λ:

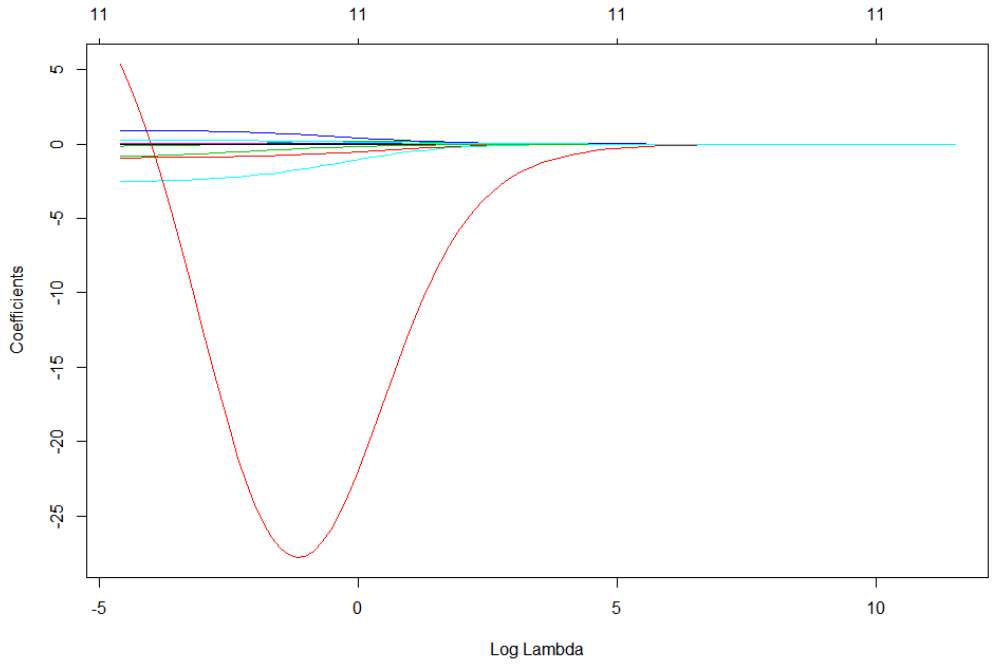
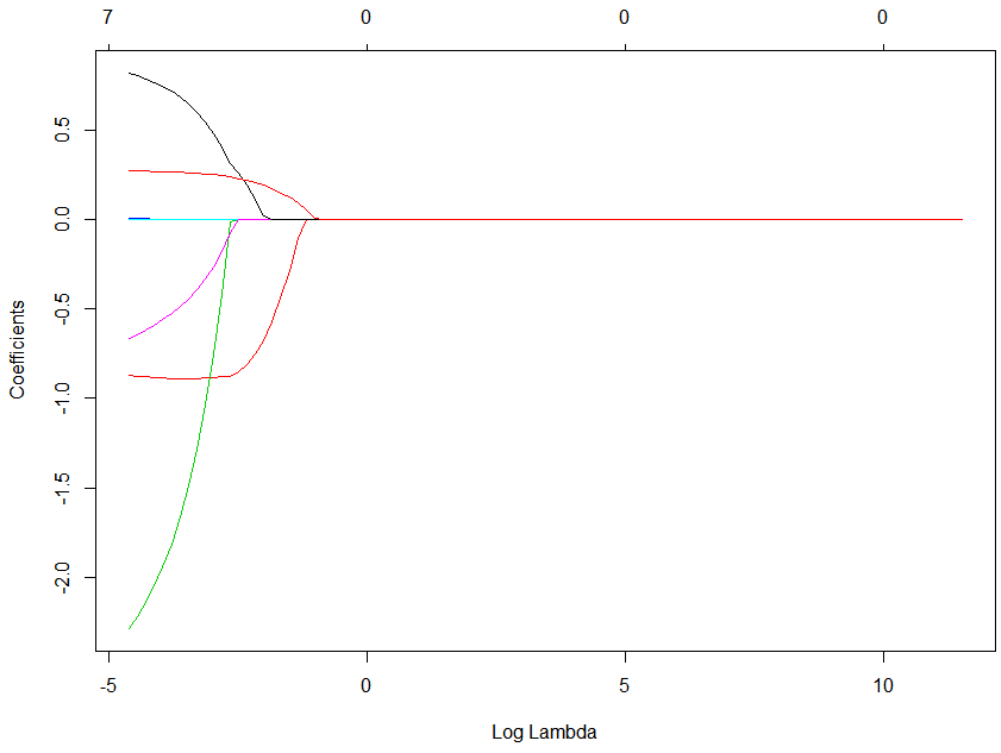


Figure 3. Coefficient estimates β for LASSO regression for the red wine data plotted versus log λ:



From the plot given in figure 2, as λ increases, the coefficient estimates β are "shrunk" towards zero as the L2 norm in Ridge regression decreases. The upper part of the plot shows the number of non-zero coefficient estimates β for a given value of log λ. As it can be seen that for Ridge this number is constant for all the λ values and equals the number of predictors in the data. Thus, although the Ridge regression shrinks the coefficient estimates close to zero, even large - values do not produce estimates exactly equal to zero.

From the plot given in figure 3, the LASSO regression also tends to "shrink" the regression coefficients to zero as λ increases. It can be noted that the numbers in the upper part of the plot which again mean the number of non-zero coefficients in the regression model. For instance, when log λ = −5 there are 7 non-zero coefficients and when log λ = 0 we get the model where all the coefficients are zero. That is, the LASSO regression performs variable selection when λ is large enough.

Now, it is the time to select the λ values for both Ridge and LASSO using cross-validation procedure. We can use `cv.glmnet` command for Ridge and LASSO which differ only by alpha values. We can also plot the results of the cross-validation results as in the figure 4 for Ridge and figure 5 for LASSO.

```
set.seed(1)
ridge.cv.out = cv.glmnet(x, train$quality, alpha=0)

set.seed(1)
lasso.cv.out = cv.glmnet(x, train$quality, alpha=1)

plot(ridge.cv.out,col="blue")
plot(lasso.cv.out,col="blue")
```

Figure 4. Cross-validated estimate of the mean squared prediction error for Ridge against log λ:

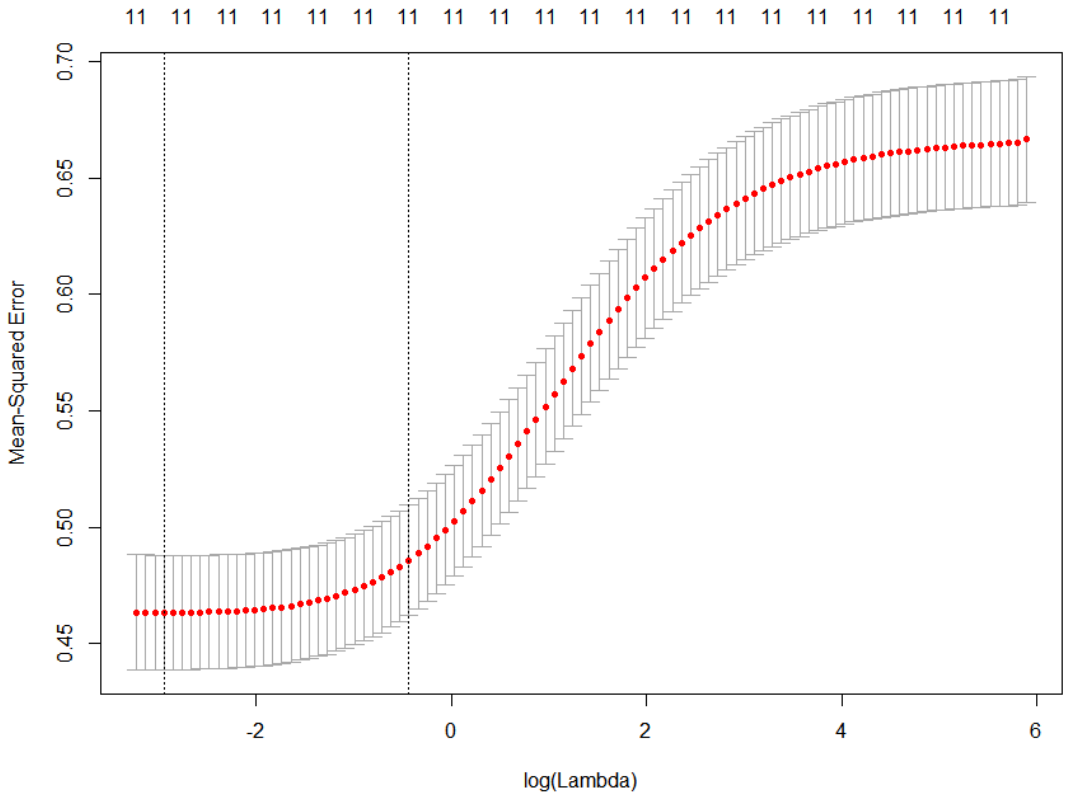
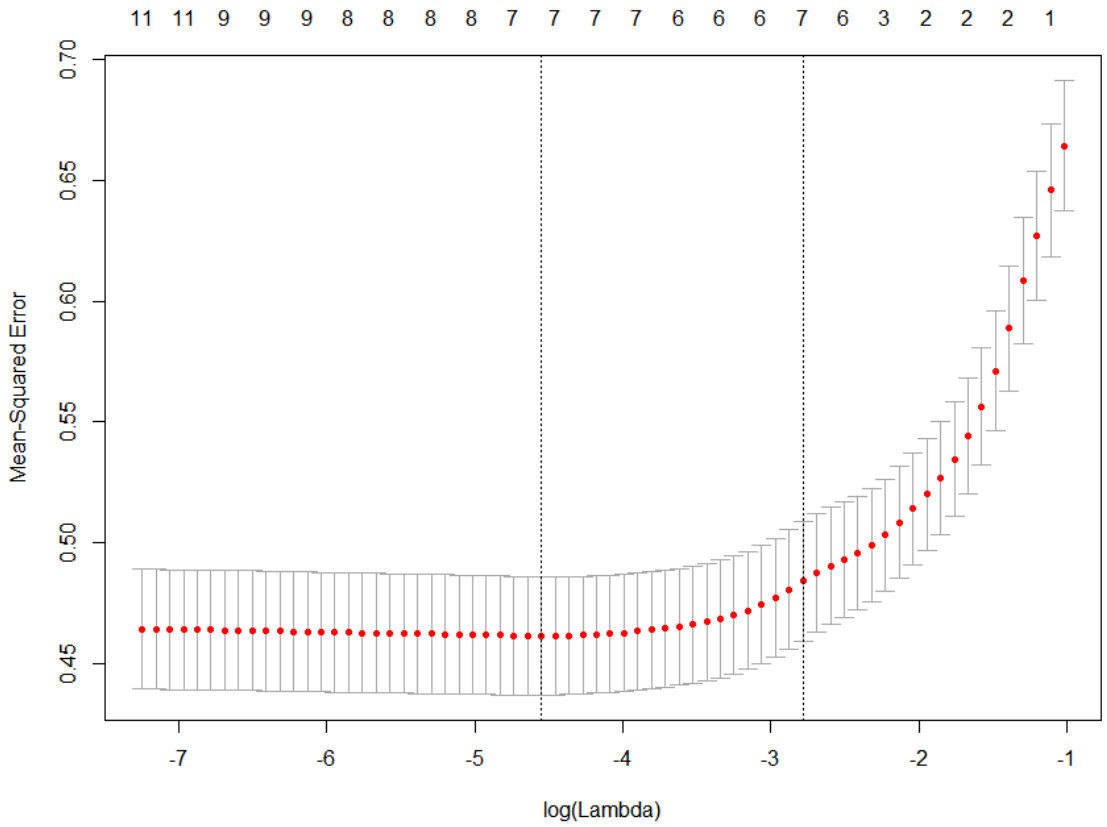


Figure 3. . Cross-validated estimate of the mean squared prediction error for LASSO against log λ:



Both these plots depict the mean squared prediction error MSE_{λ_s} against log λ. The grey bars at each point show MSE plus and minus one standard error. The first vertical dashed lines shows the location of the minimum of MSE. The second dashed line shows the point selected by the "one-standard-error" rule.

Here in our study we considered the selection of best λ for only the minimum of MSE which can be found by cross-validation results.

```
## Best lambda for Ridge
ridge.bestλ = ridge.cv.out$lambda.min
ridge.bestλ

## Best lambda for LASSO
lasso.bestλ = lasso.cv.out$lambda.min
lasso.bestλ
```

In our case, Ridge regression gives ridge.bestλ = 0.05257397 while the LASSO yields lasso.bestλ = 0.01056334 correspondingly.

The coefficient estimates computed for each of these λ values are given in the table 3. To calculate the estimates, we need to perform regression for each λ on Ridge and LASSO models.

```
## coefficients for Ridge and LASSO models using their best lambdas:

ridge.mod.best = glmnet(x, train$quality, alpha=0, lambda=ridge.bestλ)
coef(ridge.mod.best)
```

```
lasso.mod.best = glmnet(x, train$quality, alpha=1, lambda=lasso.bestλ)
coef(lasso.mod.best)
```

As it can be seen from the table 3 that both Ridge and LASSO produce a more "regularized" model as that the coefficient estimates get more "shrunk" towards zero.

Table 3. Coefficient estimates for the red wine data for different regression techniques

Method	OLS	Ridge λ=0.05257397	LASSO λ=0.01056334
Intercept terms	-8.915	18.61	5.179
fixed.acidity	-0.013	0.015	0
volatile.acidity	-0.942	-0.876	-0.874
citric.acid	-0.164	-0.053	0
residual.sugar	-0.005	0.006	0
chlorides	-2.599	-2.364	-2.264
free.sulfur.dioxide	0.007	0.005	0.004
total.sulfur.dioxide	-0.004	-0.004	-0.003
density	14.973	-13.484	0
pH	-0.929	-0.638	-0.657
sulphates	0.893	0.858	0.8122
alcohol	0.294	0.251	0.27

Let us now compare the RSS values for all the models. We will first calculate the RSS on the train set and then move to the test set.

```
# RSS on the train set and then move to the test set.
x.test <- model.matrix(quality ~ 0 + . , data=test)

# Errors in Linear model
y.lm.train = predict(m1, newdata = data.frame(x))
sum((y.lm.train - train$quality)^2)
y.lm.test = predict(m1, newdata = data.frame(x.test))
sum((y.lm.test - test$quality)^2)

# Errors in Ridge model
y.ridge.best.train = predict(ridge.mod.best, newx=x)
sum((y.ridge.best.train - train$quality)^2)
y.ridge.best.test = predict(ridge.mod.best, newx=x.test)
sum((y.ridge.best.test - test$quality)^2)

# Errors in LASSO model
y.lasso.best.train = predict(lasso.mod.best, newx=x)
sum((y.lasso.best.train - train$quality)^2)
y.lasso.best.test = predict(lasso.mod.best, newx=x.test)
sum((y.lasso.best.test - test$quality)^2)
```

Table 4. RS S values for the train (RSS train) and the test sets (RSS test) for different regression techniques.

RSS	OLS	Ridge λ=0.05257397	LASSO λ=0.01056334
RSS train	354.7427	355.846	356.2117
RSS test	318.1009	316.3684	315.7833

The error values for all the methods are given in the table 4. Here using the cross-validation procedure, the smallest RSS value on the train set is predictably achieved by the OLS regression since unlike Ridge and LASSO the OLS does not impose penalties on the coefficients β. While the case of test error, the minimum is achieved by LASSO at λ = lasso.bestλ (0.01056334), the second is Ridge at λ = ridge.bestλ (0.05257397) while OLS takes only the third position. In this case, Ridge and LASSO perform better than OLS on the test set.

4 Conclusion:

- The Ridge regression and the LASSO tend to "shrink" to zero coefficient estimates β in the sense that they reduce the norm of the estimate vector as λ increases.
- The Ridge regression does not produce zero estimates even for large values of λ.
- The LASSO, unlike Ridge and OLS, handles multicollinearity i.e. some of the coefficient estimates β become exactly equal to zero, which makes the regression model easier to interpret.
- Interesting part here in LASSO is that it not only handle multicollinearity but also performs variable selection as in the cast of "residual.sugar" variable which have VIF of 1.673 only but it has very less correlation value of 0.013 with the target variable "quality". So, adding this variable in the model has no sense and hence LASSO handles it smoothly.
- In the considered examples, when λ is selected appropriately, the RSS value on the test set for Ridge and LASSO is lesser than the RSS for OLS.
- Calculating the RSS on the test data set provides a good way to assess the regression model in contrast to using a single data set.

5 Acknowledgement:

This comparative study on Linear, Ridge and Lasso Regression models for data containing multi-collinear variables was done under the guidance of Gourab Nath, Faculty, Data Science, Praxis Business School, Bengaluru.

6 References:

- Dataset: https://archive.ics.uci.edu/ml/datasets/Wine+Quality
- https://en.wikipedia.org/wiki/Regularization_(mathematics)
- https://en.wikipedia.org/wiki/Tikhonov_regularization
- https://en.wikipedia.org/wiki/Linear_regression
- https://www.analyticsvidhya.com/blog/2017/06/a-comprehensive-guide-for-linear-ridge-and-lasso-regression/
- https://waterprogramming.wordpress.com/2017/02/22/dealing-with-multicollinearity-a-brief-overview-and-introduction-to-tolerant-methods/