Ridge Regression

Ridge regression is one of the more popular, albeit controversial, estimation procedures for combating multicollinearity.
- Raymond H. Myers

Outline

Learning Objectives

In this video, you will learn to:

- Understand the model, the cost function and the regularisation parameter $\boldsymbol{\lambda}$ of Ridge Regression.
- Learn to pre-process the data by standardisation.
- Learn to train and evaluate a Ridge Regression model in R.
- Learn to use the Cross Validation method to pick the optimal λ value.

Introduction to Ridge Regression

Cost Function for Linear Regression

Linear Regression Model

$$Y = \beta_0 + \beta_1 X_1 + \beta_1 X_2 + \cdots + \beta_n X_n$$

- The coefficients of the model are achieved via minimising the following cost function:

$$\text{Cost Function} = \sum_i \text{Residual}_i^2$$

Cost Function for Ridge Regression

Ridge Regression Model

$$Y = \beta_0 + \beta_1 X_1 + \beta_1 X_2 + \cdots + \beta_n X_n$$

- As we apply Regularisation to a MLR model, the Ridge Regression model resembles the MLR model.
- The coefficients of the model are achieved via minimising the following cost function:

$$\begin{aligned}\text{Cost Function} &= \sum_i \text{Residual}_i^2 + \boldsymbol{\lambda} \sum_{j=1}^{n} \text{Coefficients}^2 \\ &= \sum_i \text{Residual}_i^2 + \boldsymbol{\lambda} \sum_{j=1}^{n} \beta_j^2 \qquad \text{where } \lambda \geq 0\end{aligned}$$

Regularisation Parameter

$$\text{Cost Function} = \sum_i \text{Residual}_i^2 + \boldsymbol{\lambda} \sum_{j=1}^{n} \text{Coefficients}^2$$

$$= \sum_i \text{Residual}_i^2 + \boldsymbol{\lambda} \sum_{j=1}^{n} \beta_j^2 \qquad \text{where } \lambda \geq 0$$

- We will refer to λ as ***Regularisation parameter***.
- λ is manually specified, and it can be zero or any positive number.
- When $\lambda = 0$, the Ridge Regression model is same as the MLR model.
- When $\lambda = 1$, the Ridge Regression model will have smaller (predictors') coefficients, compared with the MLR model.
- In general, when λ increases, the coefficients of the Ridge Regression model will approach zero.

Introduction to Standardisation

Data Pre-processing: Standardisation

Standardisation

For any numerical variable, X, we define the ***Standardised Variable***, $\widehat{X}$ as follows,

$$\widehat{X} = X_{standardised} = \frac{X}{\sigma_X}$$

where σ_X is the standard deviation of X.

- The original MLR model is

$$Y = \beta_0 + \beta_1 X_1 + \beta_1 X_2 + \cdots + \beta_n X_n$$

- After we standardise all the numerical variables, from $X_1, X_2, \cdots, X_n, Y$ to $\widehat{X_1}, \widehat{X_2}, \cdots, \widehat{X_n}, \widehat{Y}$, we use the standardised data to train a new standardised MLR model:

$$\widehat{Y} = \hat{\beta}_0 + \hat{\beta}_1 \widehat{X_1} + \hat{\beta}_2 \widehat{X_2} + \cdots + \hat{\beta}_n \widehat{X_n}$$

where $\hat{\beta}_0, \hat{\beta}_1, \cdots, \hat{\beta}_n$ are referred to as the ***Standardised Coefficients***.

How to Interpret the Standardised Coefficient

- For each standardised predictor, $\hat{X}_i$, the standardised regression coefficient, $\hat{\beta}_i$, represents the expected change in $\hat{Y}$, due to one unit increase in $\hat{X}_i$, with all other (standardised) predictors unchanged.
- In other words, if we increase X_i by one standard unit, namely, increase X_i by σ_{X_i}, and fix all other predictors, we would expect Y change by $\hat{\beta}_i * \sigma_Y$.

- One benefit of standardisation is that we can rank the predictors based on the absolute values of the standardised coefficients.
- The larger the absolute value of the standardised coefficient, the higher importance the predictor has.

Relationship between Unstandardised and Standardised Coefficients

- The standardised coefficient, $\hat{\beta}_i$, of any predictor, X_i, can be calculated using the unstandardised coefficient of X_i, namely, β_i, first multiplied by the standard deviation of X_i, and then divided by the standard deviation of Y. The mathematical form is as follows,

$$\hat{\beta}_i = \beta_i \frac{\sigma_{X_i}}{\sigma_Y}$$

- The standardised intercept, $\hat{\beta}_0$, can be calculated using the unstandardised intercept, namely, β_0, divided by the standard deviation of Y. It can be mathematically expressed as following,

$$\hat{\beta}_0 = \frac{\beta_0}{\sigma_Y}$$

Introduction to the glmnet() Function

Introduction to the glmnet() Function

```
glmnet(x, y, alpha = 0, lambda = L)
```

Regarding the inputs:

- x is a data matrix of predictor variables.
- We will later explain how to use the function "model.matrix()" to transform any data frame into a data matrix.
- y is the dependent variable.
- Alpha is the mixing parameter. It can be any value between 0 and 1. If alpha equals:
  - "**0**": the model is trained for Ridge Regression.
  - "**1**": the model is trained for LASSO Regression.
  - Strictly between 0 and 1, e.g., 0.5, the model is trained for Elastic Net Regression.
- Lambda is the regularisation parameter.
- The input "***L***" can be any constant value that is greater than, or equal to 0. Or, "***L***" can be a sequence of values.

Assumptions of Ridge Regression Models

Assumptions of Ridge Regression Models

1. ***Independence***: Each observation is independent from the others.
2. ***Linearity***: The relationship, between the predictors Xs and the dependent variable Y, is linear.
3. ***Constant Variance***: The residuals are evenly scattered around the center line of zero.

- The least squares method provides unbiased estimates of the coefficients.
- The Ridge and LASSO Regression models output some biased estimates of the coefficients.
- P values and Confidence intervals are not meaningful for the regularised models.
- The assumption, that residuals are normally distributed, is not required.

A Case Study

Case Study: Housing Price Dataset

Story

Mr. Tan is a real estate property agent living in Boston, who wants to predict the median selling price of the houses that are located in different town areas of Boston.

He wishes to understand which factors of the area have the strongest impact on the selling price.

Source: `https://www.freepik.com/`

Ask: Formulate Focused Question

Focus Question

What is the expected selling price of houses from one neighbourhood, given the conditions and relevant factors of the area?

Source: https://www.qlik.com/blog/essential-steps-to-making-better-data-informed-decisions

Acquire: Inspect the Dataset

- Load the dataset, and check the first few observations.

```
df.housing <- read.csv("data/Boston_housing price.csv")
head(df.housing)
```

```
  Crime_rate Industry Number_of_rooms Access_to_highways Tax_rate Price
1    0.00632     2.31           6.575                  1      296  24.0
2    0.02731     7.07           6.421                  2      242  21.6
3    0.02729     7.07           7.185                  2      242  34.7
4    0.03237     2.18           6.998                  3      222  33.4
5    0.06905     2.18           7.147                  3      222  36.2
6    0.02985     2.18           6.430                  3      222  28.7
```

- Predictor variables: Crime rate, Industry, Number of rooms, Access to highways, and Tax rate.
- Dependent variable: Price.

Acquire: Inspect the Dataset

- Crime rate indicates the crime rate per capita.
- Industry is an index number that tells the proportion of the land used for non-retail business.
- Number of rooms indicates the average number of rooms per dwelling.
- Access to highways is an index measuring the accessibility to the radial highways.
- Tax rate indicates the full-value property tax rate per $10000, by town.
- For each observation, Price, in $10000s, is calculated using the median price of the houses in the same neighbourhood area.

Source: `https://www.qlik.com/blog/essential-steps-to-making-better-data-informed-decisions`

Check the Data Structure

- Check the structure of the data frame.

```
str(df.housing)
```

```
'data.frame': 506 obs. of  6 variables:
 $ Crime_rate         : num  0.00632 0.02731 0.02729 0.03237 0.06905 ...
 $ Industry           : num  2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...
 $ Number_of_rooms    : num  6.58 6.42 7.18 7 7.15 ...
 $ Access_to_highways: int  1 2 2 3 3 3 5 5 5 5 ...
 $ Tax_rate           : int  296 242 242 222 222 222 311 311 311 311 ...
 $ Price              : num  24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
```

- There are 506 observations, and all the 6 variables are numerical.

Analyse: Check for Missing or Duplicate Data

- Let us first check the missing entries, or "NA", in the dataset.

```
sum(is.na(df.housing))
```

```
[1] 0
```

- Then check the duplicate data.

```
sum(duplicated(df.housing))
```

```
[1] 0
```

- There is no missing data and no duplicate data.

Source: https://www.qlik.com/blog/essential-steps-to-making-better-data-informed-decisions

Data Pre-processing: Standardisation

- We use the apply() function to get the standard deviations of all variables uniformly, and then record the list of standard deviations to "scaler".

```
scaler <- apply(df.housing, 2, sd)
scaler
```

```
Crime_rate  Industry Number_of_rooms Access_to_highways    Tax_rate     Price
 8.6015451 6.8603529       0.7026171          8.7072594 168.5371161 9.1971041
```

- Then, we use the apply() function to apply standardisation to each variable, and record the standardised data frame to "df.housing".

```
df.housing0 <- df.housing
df.housing <- as.data.frame(apply(df.housing0, 2,
                                  function (x) x/sd(x)))
```

Data Pre-processing: Standardisation

- We can check the standard deviations of the standardised data, “df.housing”.

```
apply(df.housing, 2, sd)
```

```
Crime_rate      Industry     Number_of_rooms Access_to_highways Tax_rate     Price
         1             1                   1                  1        1         1
```

- To get the unstandardised value, we can use the standardised value, of some variable, multiplied by the standard deviation of the variable.
- For example, the unstandardised price equals, the standardised price (recorded at df.housing), multiplied by the standard deviation of price (recorded at scaler).

```
df.housing[1,6]*scaler[6]
```

```
Price
   24
```

```
df.housing0[1,6]
```

```
[1] 24
```

Analyse: Descriptive Analytics

```
summary(df.housing)
```

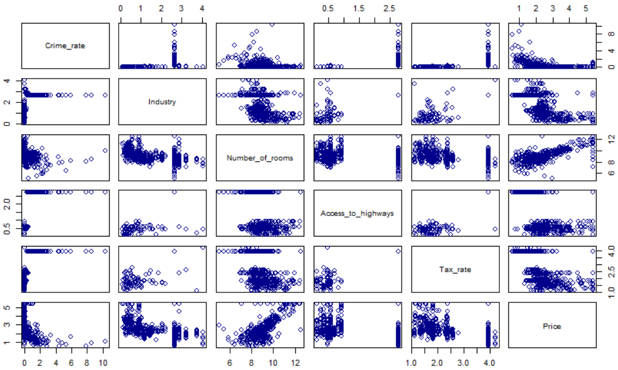
```
 Crime_rate          Industry         Number_of_rooms  Access_to_highways
 Min.    : 0.000735  Min.    :0.06705  Min.    : 5.068  Min.    :0.1148
 1st Qu.: 0.009538   1st Qu.:0.75652   1st Qu.: 8.377   1st Qu.:0.4594
 Median : 0.029821   Median :1.41246   Median : 8.836   Median :0.5742
 Mean   : 0.420102   Mean   :1.62335   Mean   : 8.945   Mean   :1.0967
 3rd Qu.: 0.427491   3rd Qu.:2.63835   3rd Qu.: 9.427   3rd Qu.:2.7563
 Max.   :10.344211   Max.   :4.04352   Max.   :12.496   Max.   :2.7563
 Tax_rate            Price
 Min.   :1.110       Min.   :0.5436
 1st Qu.:1.655       1st Qu.:1.8511
 Median :1.958       Median :2.3051
 Mean   :2.422       Mean   :2.4500
 3rd Qu.:3.952       3rd Qu.:2.7182
 Max.   :4.219       Max.   :5.4365
```

- “Crime rate” and “Access to highways”, are clearly skewed.

Analyse: Data Visualisation

- We use the “plot()” function to visualise the association between each pair of predictor variables.

```
plot(df.housing, col = "darkblue", cex = 1.2)
```



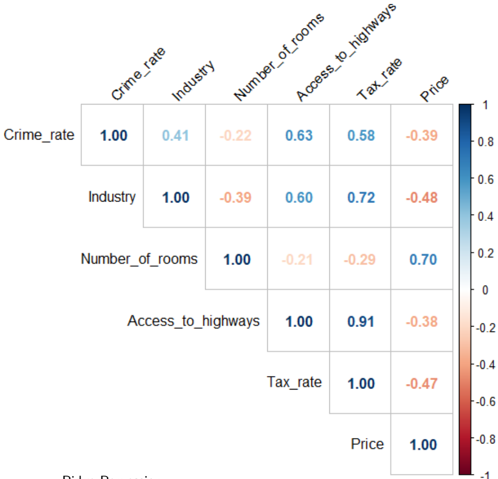
- “Number of rooms” and “Price” seem to have a strong positive correlation.

Analyse: Correlation Matrix

```
corrplot(cor(df.housing), method = "number", type = "upper",
         tl.col = "black", tl.srt = 45)
```

From the corrplot, we have the following observations:

- "Number of rooms" and "Price" are strongly and positively correlated, with r = 0.70.
- All other predictors are moderately and negatively correlated with "Price".
- "Access to highways" and "Tax rate" are strongly and positively correlated, with r = 0.91.
- "Industry" and "Tax rate" are also strongly and positively correlated, with r = 0.72.
- In conclusion, Multicollinearity seems to exist.



Multicollinearity: VIF Scores

- To check Multicollinearity, we first train a MLR model, and use vif() function to check the VIF score of each predictor.

```
lm_model0 <- lm(Price ~., data = df.housing)
vif(lm_model0)
```

```
Crime_rate        Industry     Number_of_rooms Access_to_highways    Tax_rate
  1.665394        2.326771            1.203412           6.729855    8.266513
```

- "Access to highways" and "Tax rate", with VIF scores above 5, confirm the existence of Multicollinearity.

Impact of Multicollinearity

```
summary(lm_model0)
```

From the summary table, we observe that:

- "Industry" and "Access to highways" are not significant predictors.
- The coefficient of "Access to highways" is positive, 0.13, while the correlation between "Access to highways" and "Price" is negative, with r = -0.38.
- "Number of rooms" is the most influential one, as its standardised coefficient, 0.58, has the largest absolute value among all.

```
Call:
lm(formula = Price ~ ., data = df.housing)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.8503 -0.3386 -0.0789  0.2295  4.3755 

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)    
(Intercept)         -2.02868    0.33163  -6.117 1.92e-09 ***
Crime_rate          -0.15169    0.03751  -4.044 6.09e-05 ***
Industry            -0.07290    0.04434  -1.644 0.100808    
Number_of_rooms      0.58005    0.03189  18.189  < 2e-16 ***
Access_to_highways   0.13035    0.07541   1.729 0.084516 .  
Tax_rate            -0.27684    0.08358  -3.312 0.000993 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6533 on 500 degrees of freedom
Multiple R-squared:  0.5775,	Adjusted R-squared:  0.5732 
F-statistic: 136.7 on 5 and 500 DF,  p-value: < 2.2e-16
```