Predicting Labour Wages using Ridge and Lasso Regression

STUDENT COPY

Table of Contents

[Ridge and Lasso Regression 1](#_Toc52464226)

[Read and Understand the data 2](#_Toc52464227)

[Data Pre-processing 2](#_Toc52464228)

[Train-Test Split 2](#_Toc52464229)

[Standardize the Data 2](#_Toc52464230)

[! stands for not in 3](#_Toc52464231)

[Values higher will be closer to 1 and values lower will be closer to 0 3](#_Toc52464232)

[Standardisation is done to reduce the scale of the data (rescale the data) 3](#_Toc52464233)

[Dummify the Data 3](#_Toc52464234)

[Get the data into a compatible format 3](#_Toc52464235)

[Hyper-parameter Tuning 3](#_Toc52464236)

[glmnet is one of the best and fastest tuning lib 3](#_Toc52464237)

[Choosing a lambda for Lasso Regression 3](#_Toc52464238)

[Choosing a lambda for Ridge Regression 7](#_Toc52464239)

[Building The Final Model 8](#_Toc52464240)

[Building the Final Lasso Regression Model 9](#_Toc52464241)

[Building the Final Ridge Regression Model 9](#_Toc52464242)

[Model Performance Evaluation 9](#_Toc52464243)

[Lasso Regression Model Metrics 9](#_Toc52464244)

[Ridge Regression Model Metrics 10](#_Toc52464245)

# Ridge and Lasso Regression

# Read and Understand the data

labour\_data <- read.csv("labour\_income.csv")  
str(labour\_data)

## 'data.frame': 3987 obs. of 5 variables:  
## $ wages : num 10.6 11 17.8 14 8.2 ...  
## $ education: num 15 13.2 14 16 15 13.5 12 14 18 11 ...  
## $ age : int 40 19 46 50 31 30 61 46 43 17 ...  
## $ sex : chr "Male" "Male" "Male" "Female" ...  
## $ language : chr "English" "English" "Other" "English" ...

summary(labour\_data)

## wages education age sex   
## Min. : 2.30 Min. : 0.00 Min. :16.0 Length:3987   
## 1st Qu.: 9.25 1st Qu.:12.00 1st Qu.:28.0 Class :character   
## Median :14.13 Median :13.00 Median :36.0 Mode :character   
## Mean :15.54 Mean :13.34 Mean :37.1   
## 3rd Qu.:19.72 3rd Qu.:15.10 3rd Qu.:46.0   
## Max. :49.92 Max. :20.00 Max. :69.0   
## language   
## Length:3987   
## Class :character   
## Mode :character   
##   
##   
##

# Data Pre-processing

## Train-Test Split

* Split the data into train and test

set.seed(007)  
train\_rows <- sample(x = seq(1, nrow(labour\_data), 1), size = 0.7\*nrow(labour\_data))  
train\_data <- labour\_data[train\_rows, ]  
test\_data <- labour\_data[-train\_rows, ]

## Standardize the Data

* Standardize the continuous independent variables

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

std\_obj <- preProcess(x = train\_data[, !colnames(train\_data) %in% c("wages")], method = c("center", "scale"))  
train\_std\_data <- predict(std\_obj, train\_data)  
test\_std\_data <- predict(std\_obj, test\_data)

## ! stands for not in

## Values higher will be closer to 1 and values lower will be closer to 0

# Standardisation is done to reduce the scale of the data (rescale the data)

## Dummify the Data

* Use the dummyVars() function from caret to convert sex and age into dummy variables

dummy\_obj <- dummyVars( ~ . , train\_std\_data)  
train\_dummy\_data <- as.data.frame(predict(dummy\_obj, train\_std\_data))  
test\_dummy\_data <- as.data.frame(predict(dummy\_obj, test\_std\_data))

## Get the data into a compatible format

* The functions we will be using today from the glmnet package expect a matrix as an input and not our familiar formula structure, so we need to convert our dataframes into a matrix

x\_train <- as.matrix(train\_dummy\_data[, -1])  
y\_train <- as.matrix(train\_dummy\_data[, 1])  
x\_test <- as.matrix(test\_dummy\_data[, -1])  
y\_test <- as.matrix(test\_dummy\_data[, 1])

# Hyper-parameter Tuning

* Choose an optimal lambda value for the ridge and lasso regression models by using cross validation

# glmnet is one of the best and fastest tuning lib

## Choosing a lambda for Lasso Regression

* The alpha value is 1 for lasso regression

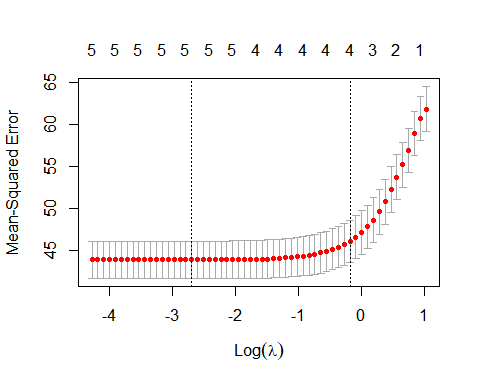
library(glmnet)

## Loading required package: Matrix

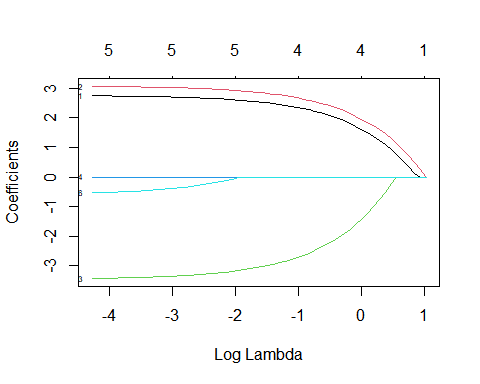
## Loaded glmnet 4.0-2

* The object returned form the call to cv.glmnet() function, contains the lambda values of importance
* The coefficients are accessible calling the coef() function on the cv\_lasso object

cv\_lasso <- cv.glmnet(x\_train, y\_train, alpha = 1, type.measure = "mse", nfolds = 4)  
plot(cv\_lasso)

 # This is a cross validation curve

plot(cv\_lasso$glmnet.fit, xvar = "lambda", label = TRUE)



print(cv\_lasso$glmnet.fit)

##   
## Call: glmnet(x = x\_train, y = y\_train, alpha = 1)   
##   
## Df %Dev Lambda  
## 1 0 0.00 2.81100  
## 2 1 2.17 2.56100  
## 3 2 4.97 2.33400  
## 4 2 8.32 2.12600  
## 5 2 11.10 1.93800  
## 6 2 13.41 1.76500  
## 7 3 15.92 1.60900  
## 8 3 18.23 1.46600  
## 9 3 20.14 1.33500  
## 10 3 21.73 1.21700  
## 11 3 23.05 1.10900  
## 12 4 24.14 1.01000  
## 13 4 25.05 0.92050  
## 14 4 25.80 0.83870  
## 15 4 26.43 0.76420  
## 16 4 26.95 0.69630  
## 17 4 27.38 0.63450  
## 18 4 27.74 0.57810  
## 19 4 28.04 0.52670  
## 20 4 28.28 0.48000  
## 21 4 28.49 0.43730  
## 22 4 28.66 0.39850  
## 23 4 28.80 0.36310  
## 24 4 28.92 0.33080  
## 25 4 29.02 0.30140  
## 26 4 29.10 0.27460  
## 27 4 29.16 0.25020  
## 28 4 29.22 0.22800  
## 29 4 29.27 0.20780  
## 30 4 29.30 0.18930  
## 31 4 29.34 0.17250  
## 32 4 29.36 0.15720  
## 33 5 29.39 0.14320  
## 34 5 29.41 0.13050  
## 35 5 29.43 0.11890  
## 36 5 29.45 0.10830  
## 37 5 29.46 0.09870  
## 38 5 29.47 0.08993  
## 39 5 29.48 0.08194  
## 40 5 29.49 0.07466  
## 41 5 29.50 0.06803  
## 42 5 29.50 0.06199  
## 43 5 29.51 0.05648  
## 44 5 29.51 0.05146  
## 45 5 29.51 0.04689  
## 46 5 29.52 0.04273  
## 47 5 29.52 0.03893  
## 48 5 29.52 0.03547  
## 49 5 29.52 0.03232  
## 50 5 29.52 0.02945  
## 51 5 29.52 0.02683  
## 52 5 29.52 0.02445  
## 53 5 29.52 0.02228  
## 54 5 29.52 0.02030  
## 55 5 29.53 0.01850  
## 56 5 29.53 0.01685  
## 57 5 29.53 0.01535  
## 58 5 29.53 0.01399

print(cv\_lasso$lambda.min)

## [1] 0.06803175

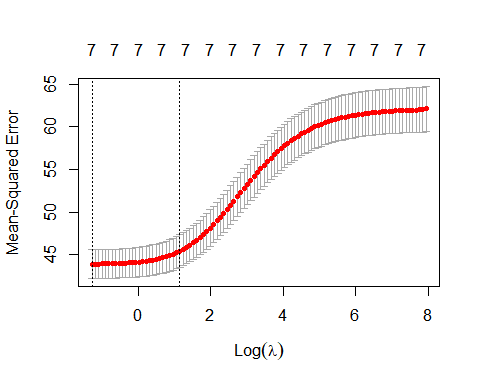
coef(cv\_lasso)

## 8 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 1.640412e+01  
## education 1.812587e+00  
## age 2.153717e+00  
## sexFemale -1.766271e+00  
## sexMale 8.259089e-14  
## languageEnglish .   
## languageFrench .   
## languageOther .

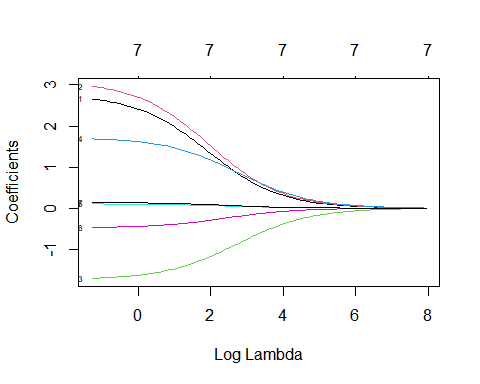
## Choosing a lambda for Ridge Regression

* The alpha value is 0 for ridge regression

cv\_ridge <- cv.glmnet(x\_train, y\_train, alpha = 0, type.measure = "mse", nfolds = 4)  
plot(cv\_ridge)



plot(cv\_ridge$glmnet.fit, xvar = "lambda", label = TRUE)



* We can access the lambda and the coefficients as we did before

print(cv\_ridge$lambda.min)

## [1] 0.281108

coef(cv\_ridge)

## 8 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 15.46625297  
## education 1.89683861  
## age 2.13566648  
## sexFemale -1.44034175  
## sexMale 1.43636353  
## languageEnglish 0.08263341  
## languageFrench -0.38150197  
## languageOther 0.11172119

# Building The Final Model

* By using the optimal lambda values obtained above, we can build our ridge and lasso models

## Building the Final Lasso Regression Model

lasso\_model <- glmnet(x\_train, y\_train, lambda = cv\_lasso$lambda.min, alpha = 1)  
coef(lasso\_model)

## 8 x 1 sparse Matrix of class "dgCMatrix"  
## s0  
## (Intercept) 1.719687e+01  
## education 2.689314e+00  
## age 3.006946e+00  
## sexFemale -3.314947e+00  
## sexMale 2.811204e-13  
## languageEnglish .   
## languageFrench -3.176708e-01  
## languageOther .

* Use the model to predict on test data

preds\_lasso <- predict(lasso\_model, x\_test)

## Building the Final Ridge Regression Model

ridge\_model <- glmnet(x\_train, y\_train, lambda = cv\_ridge$lambda.min, alpha = 0)  
coef(ridge\_model)

## 8 x 1 sparse Matrix of class "dgCMatrix"  
## s0  
## (Intercept) 15.5071089  
## education 2.6575550  
## age 2.9648092  
## sexFemale -1.7573840  
## sexMale 1.6383518  
## languageEnglish 0.1050676  
## languageFrench -0.4659695  
## languageOther 0.1378338

* Use the model to predict on test data

preds\_ridge <- predict(ridge\_model, x\_test)

# Model Performance Evaluation

## Lasso Regression Model Metrics

library(DMwR)

## Loading required package: grid

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

regr.eval(trues = y\_test, preds = preds\_lasso)

## mae mse rmse mape   
## 4.920026 43.191036 6.571989 0.380545

## Ridge Regression Model Metrics

regr.eval(trues = y\_test, preds = preds\_ridge)

## mae mse rmse mape   
## 4.9280828 43.2711221 6.5780789 0.3814506