Predicting Labour Wages using Ridge and Lasso Regression

Subiksha.S

# 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 : Factor w/ 2 levels "Female","Male": 2 2 2 1 2 1 1 1 2 2 ...  
## $ language : Factor w/ 3 levels "English","French",..: 1 1 3 1 1 1 1 3 1 1 ...

summary(labour\_data)

## wages education age sex language   
## Min. : 2.30 Min. : 0.00 Min. :16.0 Female:2001 English:3244   
## 1st Qu.: 9.25 1st Qu.:12.00 1st Qu.:28.0 Male :1986 French : 259   
## Median :14.13 Median :13.00 Median :36.0 Other : 484   
## 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

# 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)

## Warning: package 'caret' was built under R version 3.6.3

## 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)

## 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])  
library(glmnet)

## Warning: package 'glmnet' was built under R version 3.6.3

## Loading required package: Matrix

## Loaded glmnet 4.0-2

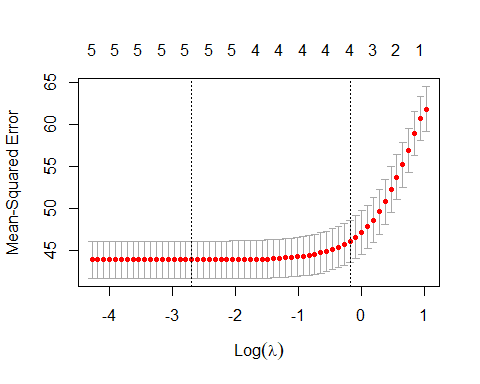
# Hyper-parameter Tuning

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

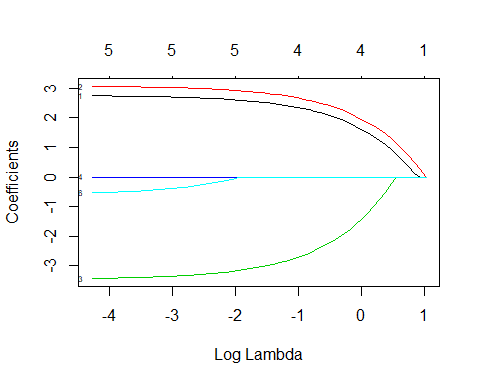
## Choosing a lambda for Lasso Regression

* The alpha value is 1 for lasso regression

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



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



* 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

print(cv\_lasso$lambda.min)

## [1] 0.06803175

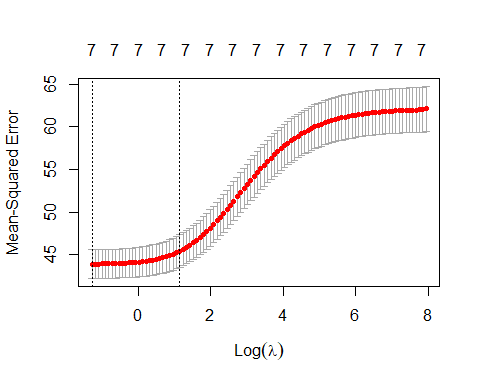
coef(cv\_lasso)

## 8 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 1.640412e+01  
## education 1.812587e+00  
## age 2.153717e+00  
## sex.Female -1.766271e+00  
## sex.Male 8.259089e-14  
## language.English .   
## language.French .   
## language.Other .

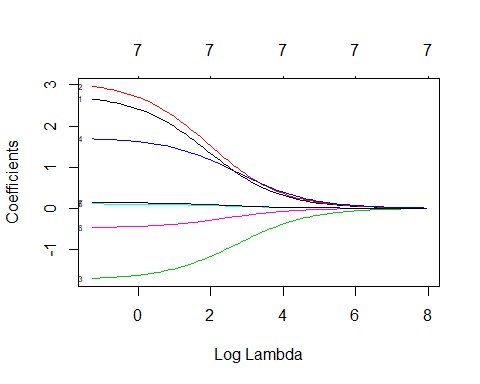
## Choosing a lambda for Ridge Regression

* 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  
## sex.Female -1.44034175  
## sex.Male 1.43636353  
## language.English 0.08263341  
## language.French -0.38150197  
## language.Other 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  
## sex.Female -3.314947e+00  
## sex.Male 2.811204e-13  
## language.English .   
## language.French -3.176708e-01  
## language.Other .

* Use the model to predict on test data

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

## 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  
## sex.Female -1.7573840  
## sex.Male 1.6383518  
## language.English 0.1050676  
## language.French -0.4659695  
## language.Other 0.1378338

* Use the model to predict on test data

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

# Model Performance Evaluation

## Lasso Regression Model Metrics

## Ridge Regression Model Metrics

library(DMwR)

## Warning: package 'DMwR' was built under R version 3.6.3

## Loading required package: grid

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

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

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