# swiss

Shiva Sankar Modala

2023-03-01

```r
# Load the swiss sample dataset from the built-in datasets (data(swiss))
data("swiss")

# Perform a basic 80/20 test-train split on the data
# Creating 80-20 Training Testing Split, createDataPartition() returns the
indices
sampleSize <- floor(0.8 * nrow(swiss))

# Setting the seed to make your partition reproducible
set.seed(123)
training_index <- sample(seq_len(nrow(swiss)), size = sampleSize)

# Training data
training_data = swiss[training_index, ]

# Testing data (note the minus sign)
testing_data = swiss[-training_index, ]

# Fitting linear model
# model_fit a linear model with Fertility as the target response,
linear_model_1 = lm(Fertility ~ ., training_data)

# What features are selected as relevant based on resulting t-statistics?
# Analyze the t-stat and p-values to select relevant features
summary(linear_model_1)

##
## Call:
## lm(formula = Fertility ~ ., data = training_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.8850  -3.0226   0.1069   3.3241  14.3459
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      67.96084   10.55947   6.436 3.57e-07 ***
## Agriculture      -0.22216    0.08167  -2.720 0.010593 *
## Examination      -0.22362    0.27124  -0.824 0.416003
## Education        -0.89779    0.18977  -4.731 4.64e-05 ***
## Catholic          0.13664    0.03446   3.965 0.000402 ***
## Infant.Mortality  1.13267    0.38546   2.939 0.006177 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.54 on 31 degrees of freedom
## Multiple R-squared:  0.7656, Adjusted R-squared:  0.7278
## F-statistic: 20.25 on 5 and 31 DF,  p-value: 6.076e-09
```

```
#  What are the associated coefficient values for relevant features?
# coefficient values for relevant features
linear_model_1$coefficients
```

```
##      (Intercept)      Agriculture      Examination        Education
##       67.9608389       -0.2221646       -0.2236157       -0.8977904
##         Catholic Infant.Mortality
##        0.1366393        1.1326696
```

```
# Predict out-of-sample
predict_out_of = predict(linear_model_1, testing_data, type = "response")
```

```
# Evaluate error
actual_data = testing_data[, "Fertility"]
cat("Out-of-Sample test MSE for regular linear model = ",
mean((predict_out_of - actual_data)^2))
```

```
## Out-of-Sample test MSE for regular linear model =  93.27207
```

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-6
```

```
# Lambda vector of 101 elements Ranging from 0 - 100000
lambda_seq = 10^seq(5, -5, by = -.1)
```

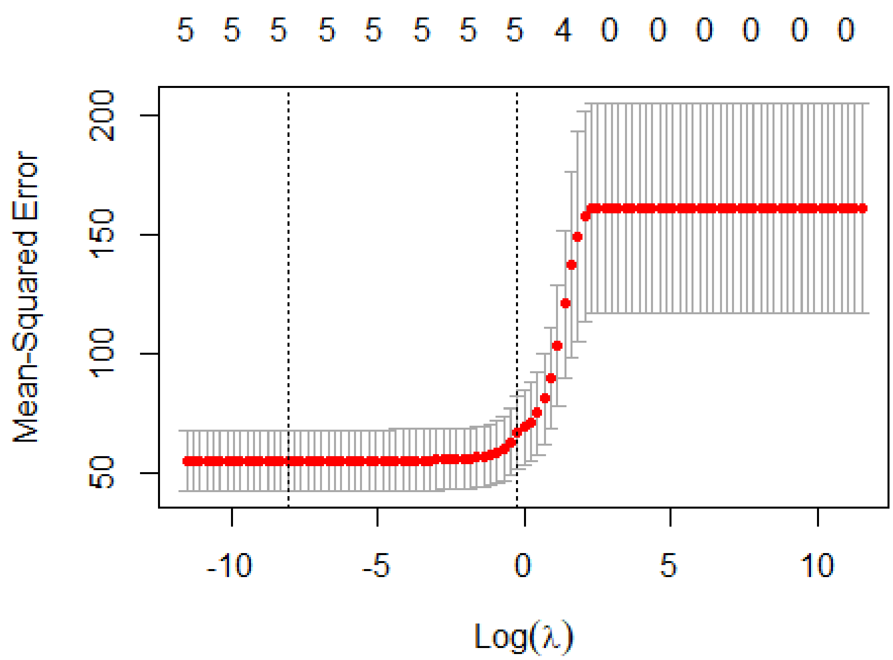
```
# Extract x and y from training data
y = training_data$Fertility
x = model.matrix(Fertility~. ,training_data)[,-1]
```

```
#  Use cross-validation (via cv.glmnet) to determine the minimum value for
lambda - what do you obtain?
# Cross-validation to perform minimum lambda
cross_validation_fit = cv.glmnet(x, y, alpha = 1, lambda = lambda_seq)
optimal_lambda = cross_validation_fit$lambda.min
cat("Optimal Lambda = ",optimal_lambda)
```

```
## Optimal Lambda =  0.0003162278
```

```
# Perform a lasso regression using the glmnet package
# Fitting Lasso Regression with optimal lambda
model_fit = glmnet(x, y, alpha = 1, lambda = optimal_lambda)
```

```r
# Plot training MSE as a function of lambda
# Plot the model
plot(cross_validation_fit)
```




```r
# Coeff. of Lasso Regression
coef(model_fit)

## 6 x 1 sparse Matrix of class "dgCMatrix"
##                          s0
## (Intercept)      67.9556030
## Agriculture      -0.2221546
## Examination      -0.2229399
## Education        -0.8981031
## Catholic          0.1366884
## Infant.Mortality  1.1324147

LassoReg_x = model.matrix(Fertility~. ,testing_data)[,-1]

# Predicting on out-of-sample test data
LassoPredict = predict(model_fit, s = optimal_lambda, newx = LassoReg_x)

# Evaluate error
actual_data = testing_data[, "Fertility"]
cat("Out-of-Sample test MSE with Lasso Regression = ", mean((LassoPredict -
actual_data)^2))

## Out-of-Sample test MSE with Lasso Regression =  93.27388
```

```r
cat ("After the Lasso, we are supposed to get some coefficient perfectly
equal to zero, however we aren't getting such results, rather the
coefficients have shrunk to some extent and the out-of-sample MSE has raised
a little bit from 93.27207 to 93.27388. Lasso usually performs variable
selection, but in this case it is performing shrinkage.")
```

```
## After the Lasso, we are supposed to get some coefficient perfectly equal
to zero, however we aren't getting such results, rather the coefficients have
shrunk to some extent and the out-of-sample MSE has raised a little bit from
93.27207 to 93.27388. Lasso usually performs variable selection, but in this
case it is performing shrinkage.
```