Example_PenalizedRegression

Lucinda Sisk

9/19/2019

Elastic Net

Elastic Net produces a regression model that is penalized with both the L1-norm and L2-norm. The consequence of this is to effectively shrink coefficients (like in ridge regression) and to set some coefficients to zero (as in LASSO).

Example using **stack overflow - click for link**

```r
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------------------------------
```

```
## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ---------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(glmnet)
```

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

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loading required package: foreach
```

```
##
## Attaching package: 'foreach'
```

```
## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```
## Loaded glmnet 2.0-18
```

```r
#Subset example data into predictor and outcome variables
data(iris)
predictor_iris <- model.matrix(~., iris)[,2:5]
outcome_iris <- iris$Species %>%
    as_factor()

# Build the model using the training set
model_iris <- cv.glmnet(x = predictor_iris,
                        y = outcome_iris,
                      family = "multinomial",
                      alpha = 1,
                      nfolds = 5,
                      intercept = FALSE)

# Best tuning parameter (smallest lambda)
(coef(model_iris, s="lambda.min"))
```

```
## $setosa
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept)   .
## Sepal.Length  .
## Sepal.Width   5.772768
## Petal.Length -2.177087
## Petal.Width  -2.853807
##
## $versicolor
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                     1
## (Intercept)  .
## Sepal.Length 1.671072
## Sepal.Width  .
## Petal.Length .
## Petal.Width  .
##
## $virginica
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept)   .
## Sepal.Length -2.530123
## Sepal.Width  -5.072816
## Petal.Length  5.458911
## Petal.Width   8.471068
```

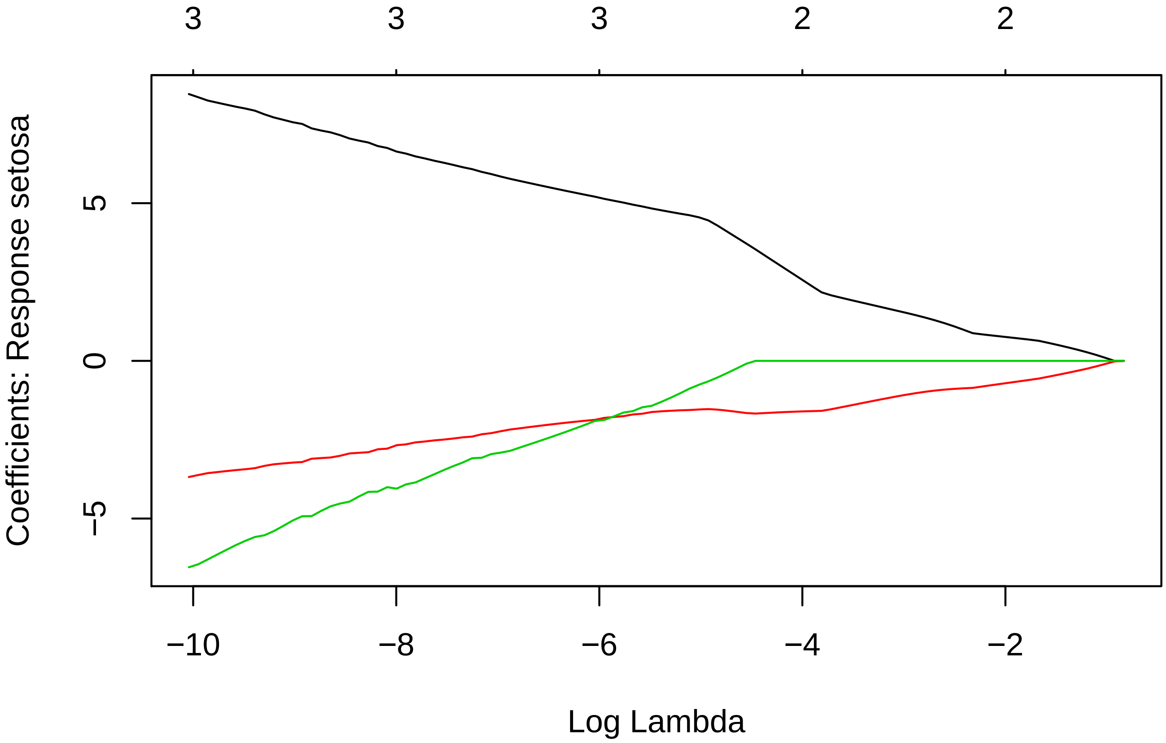
```r
#See best lambda
(model_iris$lambda[which.min(model_iris$cvm)])
```

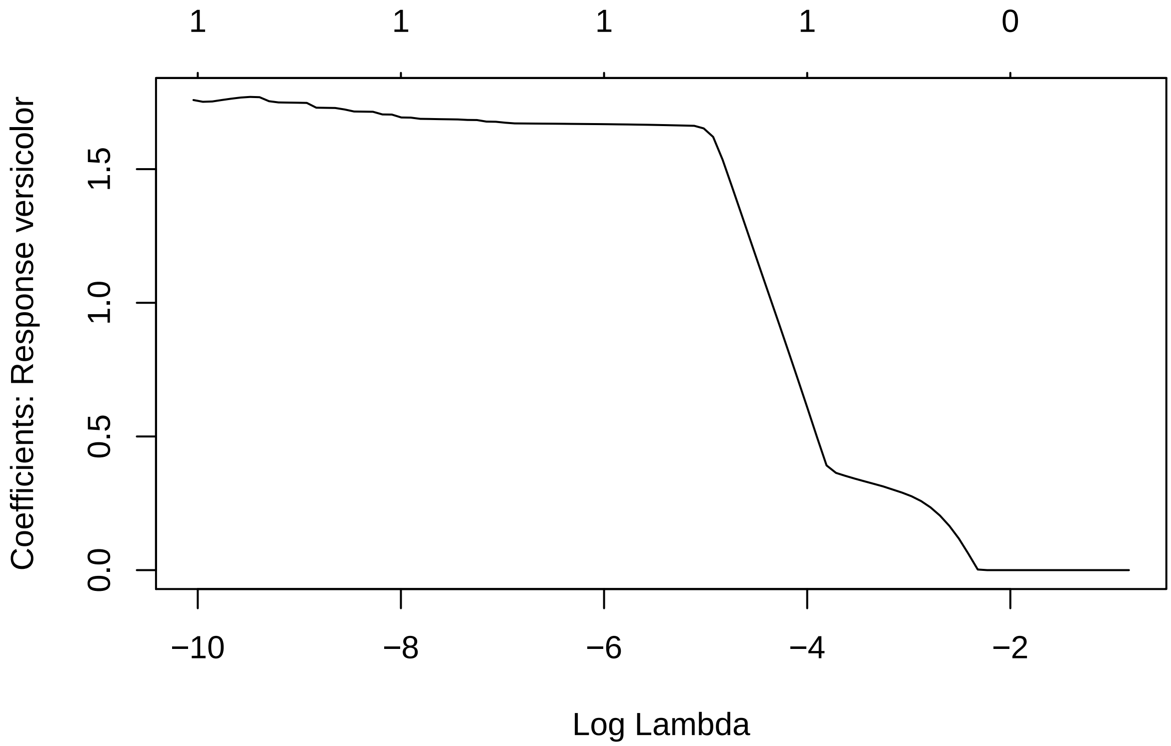
```
## [1] 0.001028543
```

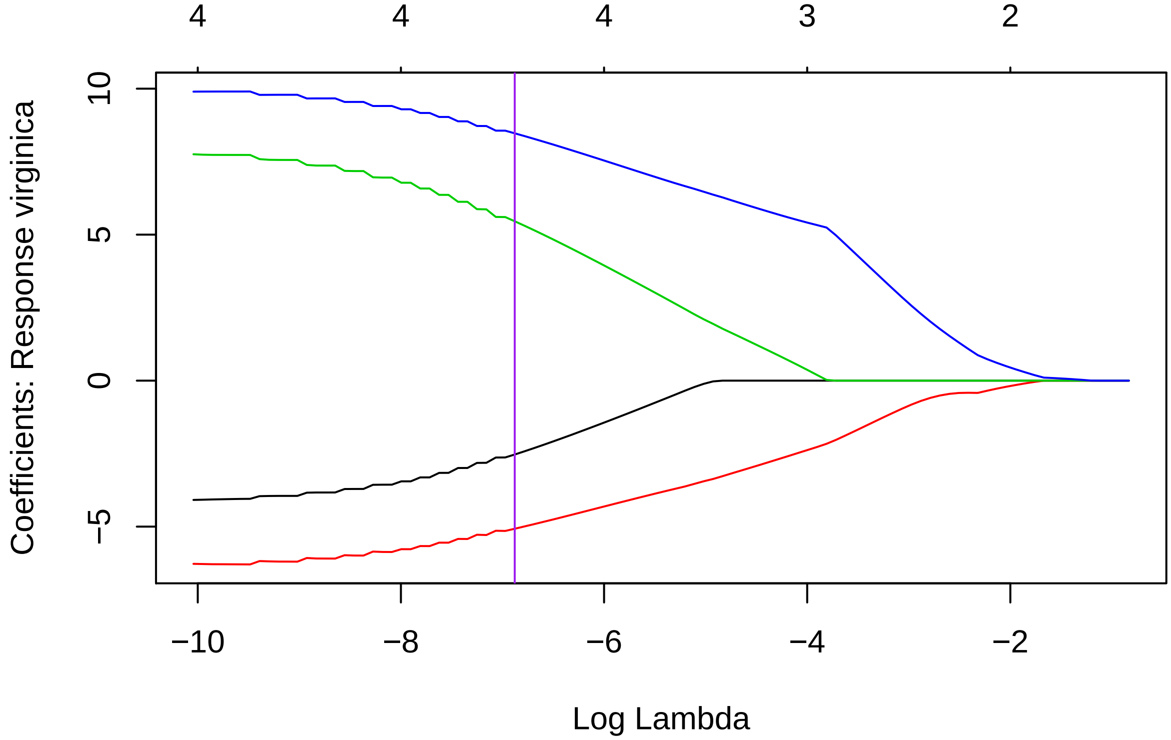
```r
#### TEST MODEL ####
#Make testing model
test_model_iris <- glmnet(x = predictor_iris,
                    y = outcome_iris,
                    family="multinomial",
                    alpha=1,
                        intercept=FALSE)

#Extract meaning from model... ?
plot(test_model_iris, xvar = "lambda") +
abline(v = log(model_iris$lambda[which.min(model_iris$cvm)]), col="purple")
```

```
## Warning in plotCoef(beta[[i]], norm, x$lambda, dfmat[i, ], x$dev.ratio, : 1
## or less nonzero coefficients; glmnet plot is not meaningful
```







```
## integer(0)
```

Extract variable names **from this Stack Overflow answer - click for link**

```
(tmp_coeffs <- coef(model_iris, s = "lambda.min"))
```

```
## $setosa
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)   .
## Sepal.Length  .
## Sepal.Width   5.772768
## Petal.Length -2.177087
## Petal.Width  -2.853807
##
## $versicolor
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  .
## Sepal.Length 1.671072
## Sepal.Width  .
## Petal.Length .
## Petal.Width  .
##
## $virginica
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)   .
## Sepal.Length -2.530123
## Sepal.Width  -5.072816
## Petal.Length  5.458911
## Petal.Width   8.471068
```

```
dimnames(coef(model_iris))
```

```
## NULL
```

Example using STHDA Resource - click for link

```
data("Boston", package="MASS")

# Split the data into training and test set
set.seed(123)
training.samples <- Boston$medv %>%
  createDataPartition(p = 0.8, list = FALSE)
train.data  <- Boston[training.samples, ]
test.data <- Boston[-training.samples, ]

# Predictor variables
predictors <- model.matrix(medv~., train.data)[,-1]
# Outcome variable
outcomes <- train.data$medv

# Build the model using the training set
set.seed(123)
model_train <- train(
  medv ~., data = train.data, method = "glmnet",
  trControl = trainControl("cv", number = 10),
  tuneLength = 10
)
# Best tuning parameter
model_train$bestTune
```

```
##   alpha     lambda
## 4   0.1 0.03875385
```

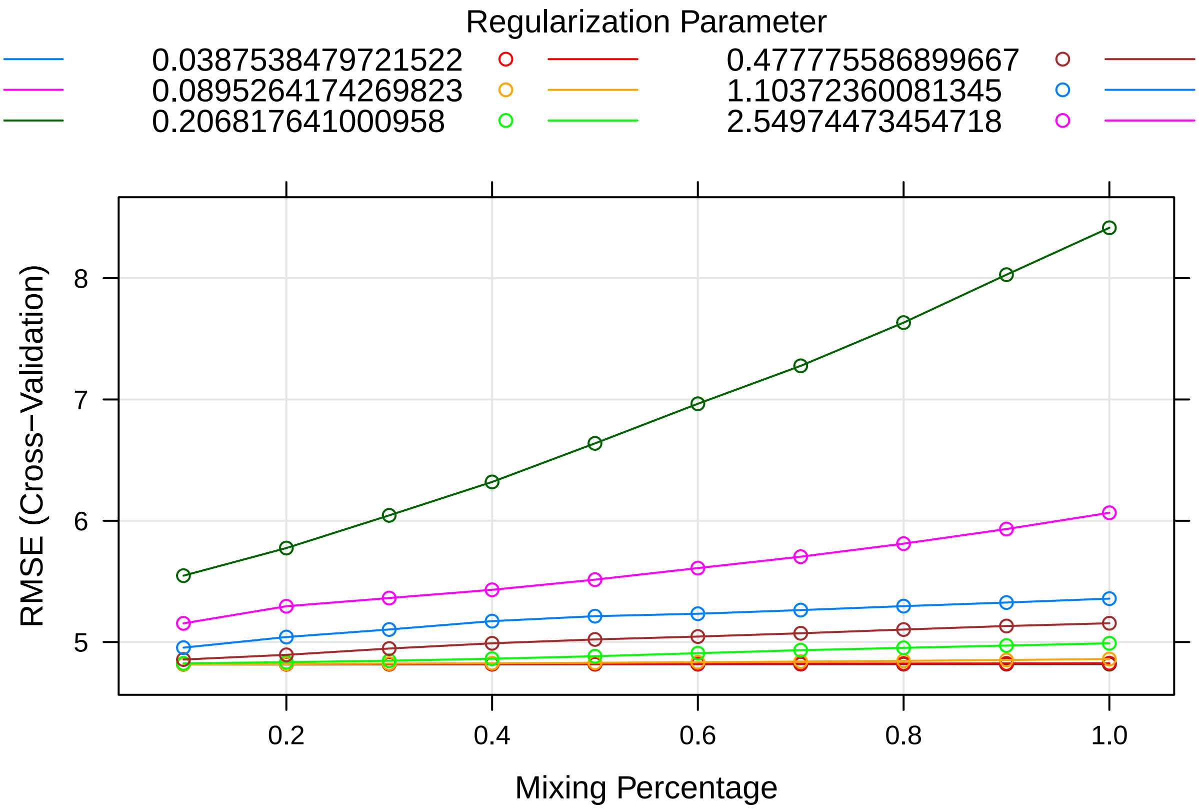
```
# Coefficient of the final model. You need
# to specify the best lambda
coef(model_train$finalModel, model_train$bestTune$lambda)
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept)  36.727488405
## crim         -0.090703096
## zn            0.037446370
## indus        -0.020415545
## chas          2.320231739
## nox         -16.457742172
## rm            3.533267286
## age           0.008434395
## dis          -1.358834790
## rad           0.305260173
## tax          -0.011175268
## ptratio      -0.950290918
## black         0.009799131
## lstat        -0.557178910
```

```
# Make predictions on the test data
x.test <- model.matrix(medv ~., test.data)[,-1]
predictions <- model_train %>% predict(x.test)
# Model performance metrics
data.frame(
  RMSE = RMSE(predictions, test.data$medv),
  Rsquare = R2(predictions, test.data$medv)
)
```

```
##       RMSE   Rsquare
## 1 4.587382 0.7621341
```

```
# Extract meaning from model???
plot(model_train, xvar = "lambda")
```



```
# coeffs <- coef(model_train$finalModel, model_train$bestTune$lambda) %>%
#     tibble()
```

Questions

1. How do we match predictors to outcome variables? I.e. is there a place to specify that subject ID links the two samples or does it figure this out automatically?
2. Do we need to pull samples from the same dataframe, two different data frames, or does it not matter?
3. We tried modeling this as in an online tutorial, but got different parameters for α and λ than the tutorial did. Is this attributable to different versions/a newer release of R? α: 0.1 λ: 0.03875385. In example, α: 0.1 λ: 0.21.
4. What is the coef function (coef(model$finalModel, model$bestTune$lambda)) doing here?

```
# Unsuccessful attempt to replicate with our data

# example_data <- read_csv('/Users/lucindasisk/Box/LS_Folders/CANDLab_LS/TraumaData/CTQ_ROI_SCR_SOBP201
# outcome_vars <- example_data %>%
#     select(Shapes2_Aminus) %>%
#     as_vector() %>%
#     replace_na(0)
#
# predictor_vars <- example_data %>%
#     select(Sex_Final, Age_Final,
#            Hindy_L_AntHipp_test1_ABminus_predot:Hindy_L_AntHipp_test1_AllShapes_dot) %>%
#     replace_na(list(x = 0, y = 0))
#
# predictor_vars2 <- model.matrix( ~ ., predictor_vars)
```