Cross Validated

Feature Importance in each fold and repeat after repeated cross validation in caret

Asked 7 years ago Active 7 years ago Viewed 1k times

1

2

this is my first post on Cross Validated so I apologize in advance if I'm not yet familiar with any conventions regarding forum posts. Currently, I'm working on a feature selection task using elastic net in caret and I would like to visualize the feature importance for each model trained in each cross validation step. However I don't seem to find a way to access the coefficients, other than those of the final model.

Here is more or less a minimal example of what I'm doing.

```
library(caret)
library(doMC)

registerDoMC(16) # register 16 cores
```

Define tuning grid of elastic net parameters.

```
grid <- expand.grid(.lambda = seq(0, 1, length=20),
                    .alpha = seq(0, 1, length = 11))
```

Provide control object defining repeated cross validation resulting in 5-fold cross validation each repeated 5 times.

```
ctrl <- trainControl(method = "repeatedcv",  # cross-validation method
                     number = 5,             # number of folds
                     repeats = 5,            # number of complete sets of folds
                     allowParallel = TRUE)   # utilize parallelization
```

Train models with defined cross validation scheme and parameter grid Furthermore the featuers will be centered and scaled.

```
model <- train(x = iris[,-5],
               y = iris$Species,
               method = "glmnet",
               type.gaussian = "naive",
               tuneGrid = grid,
               trControl = ctrl,
               preProc = c("center", "scale"))
```

Alright, now I can get some information about the test performance after repeated cross validation.

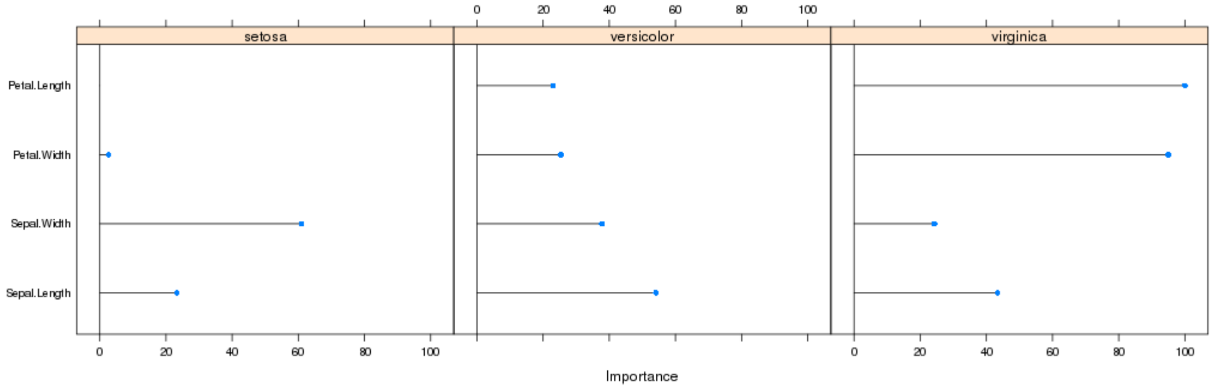
```
model$resample[with(model$resample, order(Resample)), ]

Accuracy Kappa   Resample
12 1.0000000  1.00 Fold1.Rep1
19 1.0000000  1.00 Fold1.Rep2

25 1.0000000  1.00 Fold1.Rep3
2  0.9666667  0.95 Fold1.Rep4
9  0.9333333  0.90 Fold1.Rep5
1  0.9000000  0.85 Fold2.Rep1
8  0.9333333  0.90 Fold2.Rep2
15 0.9666667  0.95 Fold2.Rep3
22 0.9333333  0.90 Fold2.Rep4
16 0.9333333  0.90 Fold2.Rep5
18 1.0000000  1.00 Fold3.Rep1
11 0.9666667  0.95 Fold3.Rep2
5  1.0000000  1.00 Fold3.Rep3
3  1.0000000  1.00 Fold3.Rep4
6  1.0000000  1.00 Fold3.Rep5
23 1.0000000  1.00 Fold4.Rep1
7  1.0000000  1.00 Fold4.Rep2
10 1.0000000  1.00 Fold4.Rep3
17 0.9333333  0.90 Fold4.Rep4
24 1.0000000  1.00 Fold4.Rep5
13 0.9333333  0.90 Fold5.Rep1
20 0.9666667  0.95 Fold5.Rep2
14 0.9333333  0.90 Fold5.Rep3
21 1.0000000  1.00 Fold5.Rep4
4  1.0000000  1.00 Fold5.Rep5
```

However, I don't see how to access the coefficients for the models generating the respective cv accuracies, to visualize the variable importance in the same way it is possible for the final model.

```
plot(varImp(model))
```



I would very much appreciate your help.

r feature-selection caret elastic-net

Share Cite Edit Follow Flag

asked Sep 10 '14 at 9:20

jhooge

141 1 9

1 Answer

Active Oldest Votes

2

`train` doesn't save the model information within a fold. You can save the models out to the file system using a custom model:

```
glmn_funcs <- getModelInfo("glmnet", regex = FALSE)[[1]]
glmn_funcs$fit <- function(x, y, wts, param, lev, last, classProbs, ...) {
    theDots <- list(...)
    if(all(names(theDots) != "family")) theDots$family <- "multinomial"
    modelArgs <- c(list(x = as.matrix(x), y = y, alpha = param$alpha),
                        theDots)

    out <- do.call("glmnet", modelArgs)
    if(!is.na(param$lambda[1])) out$lambdaOpt <- param$lambda[1]
        save(out, file = paste("~/tmp/glmn", param$alpha,
                          floor(runif(1, 0, 1)*100), ## to help uniqueness
                          format(Sys.time(), "%H_%M_%S.RData"),
                          sep = "_")
    out
  }

model <- train(x = iris[,-5],
               y = iris$Species,
               method = glmn_funcs,
               type.gaussian = "naive",
               tuneGrid = grid,
               trControl = ctrl,
               preProc = c("center", "scale"))
```

You can use the `coef` function on each model to get the slopes. Note that `train` did not fit all possible models, which is

```
> length(model$control$index)*nrow(grid)
[1] 5500
```

(omitting the one for the final model). It fits one per unique alpha per fold:

```
> length(unique(grid$.alpha))*length(model$control$index)
[1] 275
> length(list.files("~/tmp", pattern = "glmn_")) ##includes the final model
[1] 276
```

So you will have to do some looping using something like:

```
> params <- coef(out, s = unique(grid$.lambda), type = "nonzeo")
    > names(params) ## a matrix per class
    [1] "setosa"     "versicolor" "virginica"
    > lapply(params, dim)
    $setosa
[1]  5 20

$versicolor
[1]  5 20

$virginica
```

```
[1]  5 20
```

Lastly, you don't need to prefix a period before the parameter names using recent versions of `caret`.

Max

Share Cite Edit Follow Flag

answered Sep 10 '14 at 17:12

topepo

5,760 1 18 24