Lab Report Ridgereg

Simon Jönsson, Fanny Karelius

2017-10-16

Here we instantiate the training and test data with the training data being 80% and test data being 20% of the data set.

```
trainIndex <-
  caret::createDataPartition(BostonHousing$crim,
                             p = 0.8,
                             times = 1,
                             list = FALSE)
trainDat <- BostonHousing[trainIndex, ]
testDat <- BostonHousing[-trainIndex, ]
form <- tax ~ .
```

We now fit the data to the models using linear regression and linear regression with forward selection on the covariates.

```
linMod <- caret::train(form,
                       trainDat,
                       method = "lm",
                       trControl = trainC)
linFMod <- caret::train(form,
                        trainDat,
                        method = "leapForward",
                       trControl = trainC)
```

We evaluate the models with analyzing the RMSE and R^2 values.

```
linMod$results$RMSE
```

```
## [1] 54.07378
```

```
linFMod$results$RMSE
```

```
## [1] 57.07212 56.18481 54.11274
```

Analyzing the RMSE and the R^2 of the models entails some information. Lower RMSE value would indicate a 'tighter fit' of the data and a higher R^2 value indicates a measure of how well observed outcomes are replicated by the model, based on the proportion of total variation of outcomes explained by the model.

```
set.seed(theSeed)
res <- c()
for(lambda in seq(0, 20, by = 1)) {
  temp <- caret::train(form,
                       data = trainDat,
                       ridgeMod)
  res[lambda * 10] <- temp$results$RMSE
}
bestLambda <- which.min(res) / 10
bestRMSE <- res[which.min(res)]
bestLambda
```

```
## [1] 10
```

```
bestRMSE
```

```
## [1] 406.4993
```

We see from the code above that the best $\lambda = 0.5$ with the lowest RMSE of 1320.828.

```
# Acknowledgement to Eric Herwin and Albin Vasterlund
set.seed(theSeed)
fold_count <- 10
lambda <- seq(10,20,by=1)
fitControl <- caret::trainControl(method = "repeatedcv",
                                  number = fold_count,
                                  ## repeated ten times
                                  repeats = fold_count)
ridgeMod <- caret::train(form,
                         data = trainDat,
                         method = ridgeMod,
                         trControl = fitControl)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info =
## trainInfo, : There were missing values in resampled performance measures.
```

```
ridgeMod
```

```
## 406 samples
##  13 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 366, 365, 364, 366, 366, 366, ...
## Resampling results across tuning parameters:
##
##   lambda  RMSE      Rsquared   MAE
##   10      410.1854  0.8887925  406.335
##   11      410.1892  0.8887097  406.335
##   12      410.1934  0.8886175  406.335
##   13      410.1980  0.8885163  406.335
##   14      410.2030  0.8884068  406.335
##   15      410.2084  0.8882895  406.335
##   16      410.2142  0.8881647  406.335
##   17      410.2202  0.8880330  406.335
##   18      410.2266  0.8878946  406.335
##   19      410.2333  0.8877500  406.335
##   20      410.2402  0.8875995  406.335
##
## RMSE was used to select the optimal model using  the smallest value.
## The final value used for the model was lambda = 10.
```

Last line says the best value for $\lambda = 17$. Which is the same as as we concluded above, though with different RMSE. The interval for $\lambda \in 10 : 20$ is motivated by previous larger interval, but for output niceties we chose to limit lambda.

```
lin <- list(RMSE = linMod$results$RMSE, RSquared = linMod$results$Rsquared)
linF <- list(RMSE = linFMod$results$RMSE[3], RSquared = linFMod$results$Rsquared[3])
ridgeM <- list(RMSE = ridgeMod$results$RMSE[7], RSquared = ridgeMod$results$Rsquared[7])
temp <- cbind(lin,linF,ridgeM)
colnames(temp) <- c("Lin. Reg", "Lin. Reg. F", "Ridge Reg.")
```

```
temp
```

```
##          Lin. Reg  Lin. Reg. F Ridge Reg.
## RMSE     54.07378  54.11274    410.2142
## RSquared 0.8895151 0.8912715   0.8881647
```