Weekly Summary Template

Brady Miller

Table of contents

Tuesday, March 14

! TIL

Include a *very brief* summary of what you learnt in this class here.
Today, I learnt the following concepts in class:

1. More with k-fold cross validation
2. Using caret package with cross validation
3. Using caret package with LASSO regression

```
# opening some necessary libraries
library(ISLR2)
library(dplyr)
```

```

Attaching package: 'dplyr'


The following objects are masked from 'package:stats':

    filter, lag
```

```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
library(tidyr)
library(purrr)
library(readr)
library(glmnet)
```

```
Loading required package: Matrix

Attaching package: 'Matrix'

The following objects are masked from 'package:tidyr':

    expand, pack, unpack

Loaded glmnet 4.1-6
```

```r
library(caret)
```

```
Loading required package: ggplot2

Loading required package: lattice

Attaching package: 'caret'

The following object is masked from 'package:purrr':

    lift
```

```r
library(car)
```

```
Loading required package: carData

Attaching package: 'car'
```

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

    some

The following object is masked from 'package:dplyr':

    recode
```

```
library(torch)
```

```
# opening Boston data set which will be used in examples
df <- Boston
attach(Boston)
```

k-fold cross validation

```
k <- 5
folds <- sample(1:k, nrow(df), replace = T)

df_folds <- list()

# define list of data frame where every list has train and test
for (i in 1:k){
  df_folds[[i]] <- list()
  df_folds[[i]]$train = df[which(folds == i), ]
}
```

```
# finding the mean squared error for each fold variation
kfold_mspe <- c()
for (i in 1:k) {
  model <- lm(medv ~ ., df_folds[[i]]$train)
  y_hat <- predict(model, df_folds[[i]]$test)
  kfold_mspe[i] <- mean((y_hat - df_folds[[i]]$test$medv)^2)
}
```

```
make_folds <- function(df, k){
  folds <- sample(1:k, nrow(df), replace = T)
  df_folds <- list()
  for (i in 1:k){
    df_folds[[i]] <- list()
```

```r
    df_folds[[i]]$train = df[which(folds != i), ]
    df_folds[[i]]$test = df[which(folds == i), ]
  }
  return(df_folds)
}
```

```r
# cross validation mean squared prediction error function
cv_mspe <- function(formula, df_folds){
  kfold_mspe <- c()
  # going through each fold to generate predictions and find the error for each
  for (i in 1:length(df_folds)){
    model <- lm(formula, df_folds[[i]]$train)
    y_hat <- predict(model, df_folds[[i]]$test)
    kfold_mspe <- mean((y_hat - df_folds[[i]]$test$medv)^2)
  }
  return(mean(kfold_mspe))
}
```

- This function can then be used to find the prediction error generated for a given set of variables over a certain amount of folds. This can help you determine which set of variables would be best to create the least error

cross validation with caret package

By using the caret package, we can write code to cross validate a dataset in much fewer lines as shown below

```r
# specifying you want to cross validate using 5 folds (the number)
ctrl <- trainControl(method = 'cv', number = 5)
ctrl
```

```
$method
[1] "cv"

$number
[1] 5

$repeats
[1] NA

$search
```

```
[1] "grid"

$p
[1] 0.75

$initialWindow
NULL

$horizon
[1] 1

$fixedWindow
[1] TRUE

$skip
[1] 0

$verboseIter
[1] FALSE

$returnData
[1] TRUE

$returnResamp
[1] "final"

$savePredictions
[1] FALSE

$classProbs
[1] FALSE

$summaryFunction
function (data, lev = NULL, model = NULL)
{
    if (is.character(data$obs))
        data$obs <- factor(data$obs, levels = lev)
    postResample(data[, "pred"], data[, "obs"])
}
<bytecode: 0x00000228c2528d80>
<environment: namespace:caret>

$selectionFunction
```

```
[1] "best"

$preProcOptions
$preProcOptions$thresh
[1] 0.95

$preProcOptions$ICAcomp
[1] 3

$preProcOptions$k
[1] 5

$preProcOptions$freqCut
[1] 19

$preProcOptions$uniqueCut
[1] 10

$preProcOptions$cutoff
[1] 0.9


$sampling
NULL

$index
NULL

$indexOut
NULL

$indexFinal
NULL

$timingSamps
[1] 0

$predictionBounds
[1] FALSE FALSE

$seeds
[1] NA
```

```
$adaptive
$adaptive$min
[1] 5

$adaptive$alpha
[1] 0.05

$adaptive$method
[1] "gls"

$adaptive$complete
[1] TRUE


$trim
[1] FALSE

$allowParallel
[1] TRUE
```

```r
# creating model that uses linear regression model to predict med. house price
# trControl attribute uses the ctrl object to specify 5-fold cross validation
model <- train(medv ~ ., data = df, method = 'lm', trControl = ctrl)
summary(model)
```

```

Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
     Min       1Q   Median       3Q      Max 
-15.1304  -2.7673  -0.5814   1.9414  26.2526 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  41.617270   4.936039   8.431 3.79e-16 ***
crim         -0.121389   0.033000  -3.678 0.000261 ***
zn            0.046963   0.013879   3.384 0.000772 ***
indus         0.013468   0.062145   0.217 0.828520    
chas          2.839993   0.870007   3.264 0.001173 ** 
nox         -18.758022   3.851355  -4.870 1.50e-06 ***
rm            3.658119   0.420246   8.705  < 2e-16 ***
```

```
age           0.003611   0.013329   0.271 0.786595    
dis          -1.490754   0.201623  -7.394 6.17e-13 ***
rad           0.289405   0.066908   4.325 1.84e-05 ***
tax          -0.012682   0.003801  -3.337 0.000912 ***
ptratio      -0.937533   0.132206  -7.091 4.63e-12 ***
lstat        -0.552019   0.050659 -10.897  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.798 on 493 degrees of freedom
Multiple R-squared:  0.7343,    Adjusted R-squared:  0.7278 
F-statistic: 113.5 on 12 and 493 DF,  p-value: < 2.2e-16
```

This model created by cross validation using the caret package indicates that the age and indus variables are not significant in predicting the median house price of a given home. The R^2 value indicates that there is a somewhat strong positive correlation between the covariates and the outcome variable.

```
# creates predictions for each piece of data in the Boston data set
predictions <- predict(model,df)
sample(predictions,10)
```

```
     162      140      214       64      171      396      390       65 
36.73957 16.05755 25.15434 22.34303 23.19122 19.37022 13.29083 23.35022 
     141      362 
13.09512 18.45362 
```

Above we show 5 randomly selected predictions made on the data set. Each row number has the associated predicted median house value with it.

LASSO regression with caret package

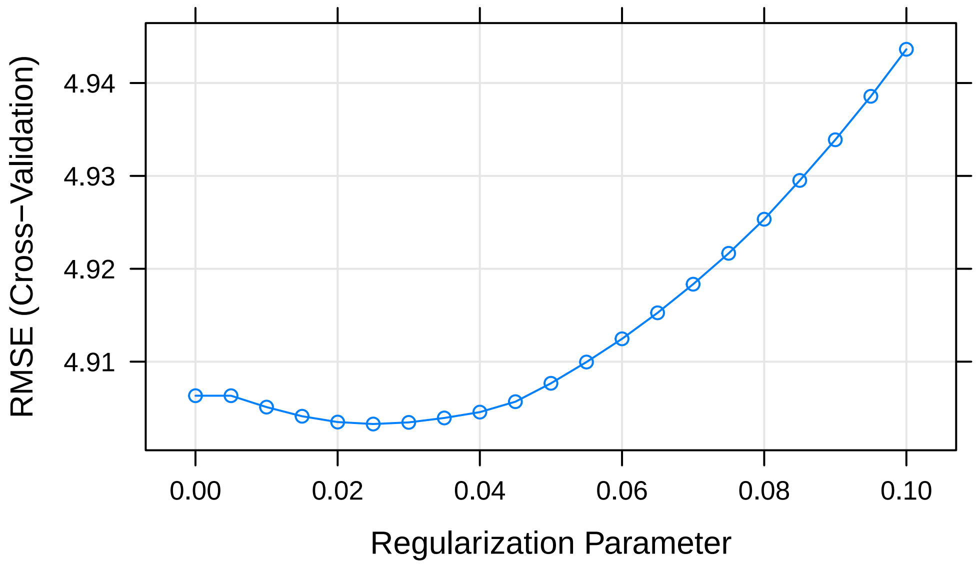
This deals with bias-variance trade off. As it states there is a trade off between having more bias or more variance. By having a model that has more variables, bias decreases but variance increases. Having too much variance can lead to overfitting data and create good performance on train data but bad performance on test data. On the other hand, a model with only a few variables increases bias and reduces variance. Increasing bias can lead to creating a model that underfits data, resulting in poor performance on both the train and test data. This relates to LASSO as it is a form of variable selection, so you want to make sure you are selecting the right amount of variables so you have a balanced trade off between bias and variance

```
ctrl <- trainControl(method = 'cv', number = 5)

# Defining the tuning grid
grid <- expand.grid(alpha = 1, lambda = seq(0, 0.1, by = 0.005))

# Train the model using LASSO regression with cross validation
lasso_fit <- train(medv ~ ., data = df, method = 'glmnet', trControl = ctrl,
                   tuneGrid = grid, standardize = TRUE, family = 'gaussian')

plot(lasso_fit)
```



This plot shows the mean squared error value for each regularization parameter tried in the LASSO regression with the cross validation. From this, we can see that the regularization parameter that minimizes the mean squared error value is at about 0.03, so λ should be 0.03 when doing LASSO regression.

Thursday, March 16

! TIL

Include a *very brief* summary of what you learnt in this class here.
Today, I learnt the following concepts in class:

1. Item 1
2. Item 2
3. Item 3