Generalized Additive Models with Cross-Fold Validation using the Advertising Data

Chris Schmidt

2019

K-Fold Validation on GAM's

This project builds generalized additive models (GAMs) using smoothing splines on the attributes of interest using the csv data in the 'Advertising.csv' file that we need to import.

Generalized additive models extend the linear model by allowing non-linear functions of each of the variables while maintaining *additivity*. GAMs can use quantitative and qualitative responses like linear models.

Given a multiple linear regression model

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip} + \epsilon_i$$

we can allow for non-linear relationships between each attribute and the response by replacing each linear component $\beta_j x_{ij}$ with a smooth non-linear function $f_j(x_{ij})$ which produces a model form

$$\begin{aligned} &= \beta_0 + \sum_{j=1}^{p} fj(x_{ij}) + \epsilon_i \\ &= \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + ...f_p(x_{ip}) + \epsilon_i \end{aligned}$$

There are a number of pros and cons to GAMs that Hastie and Tibshirani discuss in their book, *The Elements of Statistical Learning* discussed in lecture that are too broad for this assignment.

Project setup and libraries

We need the `mgcv` package.

The Advertisement data

```
adv <- read.csv('Advertising(3).csv', header = T)
dim(adv)
```

You need save Advertising.csv in the same folder as this rmd file

```
## [1] 200   5
```

Fit two models to the data. Note they differ on the degrees of freedom for TV.

Below we fit two models to the data using the **gam()** function with the smoothing spline function from the **gam** library, **s()**, and the tensor product interaction function, **ti()** applied to the attributes of interest in each model.

```
library(mgcv)

m_sr <- gam(Sales ~ s(TV) + s(Radio) + ti(TV, Radio), data = adv)

m2_sr <- gam(Sales ~ s(TV, k = 20) + s(Radio) + ti(TV, Radio), data = adv)
```

Perform 10-fold cross validation for model m_sr and m2_sr and find which model has a smaller mean absolute error. (refer to the handout 'R code for the advertisement data.txt')

k-fold cross validation is a resampling technique that randomly divides the data set into k groups of observations or roughly equal size. The first fold is the validation set and the model method is run on the remaining k-1 folds of data. The MSE_1 (mean squared error) is computed on the observations on the held-out fold and the procedure is repeated k-times with a different held-out fold as the validation set. This generates k estimates of the test error, $MSE_1, MSE_2, ..., MSE_k$. The k-fold CV (cross validation) estimate is then generated by averaging these values.

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} MSE_i$$

There is a bias-variance trade-off associated with the choice of k in k-fold cross-validation. Typically a k-value of 5 or 10 is used as these have been shown empirically to generate test-error rate estimates that do not have either high bias or high variance.

When randomly dividing the set of observations into 10 groups we use set.seed(1) to fix the random number generator so that we get the same result each time we generate the output..

```
set.seed(1)
train <- sample (200,200)
MSE <- c()
MAE <- c()
mse <- c()
mae <- c()

for (i in 1:10){
     train_data <- adv[-((i*20-19):(i*20)), ]
     test_data <- adv[((i*20-19):(i*20)), ]
     #m_sr <- gam(Sales ~ s(TV) + s(Radio) + s(Newspaper), data = train_data)
     m_sr <- gam(Sales ~ s(TV) + s(Radio) + ti(TV, Radio), data = train_data)
     mse <- c(mse, mean((predict(m_sr, newdata = test_data) - test_data$Sales)^2))

     test_pre <- predict(m_sr, newdata = test_data)
     mae <- c(mae, mean(abs(test_pre - test_data$Sales)))
}

MSE_m_sr <- c(MSE, mean(mse))
MAE_m_sr <- c(MAE, mean(mae))
MSE_m_sr
```

10-fold cross validation for model m_sr and MAE output

```
## [1] 0.2109618
```

```
summary(MSE_m_sr)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.211   0.211   0.211   0.211   0.211   0.211
```

```
MAE_m_sr
```

```
## [1] 0.2968779
```

```
summary(MAE_m_sr)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2969  0.2969  0.2969  0.2969  0.2969  0.2969
```

The mean absolute errors (MAE) for the test set for each of the 10 folds in the k-fold cross validation for model m_sr are generated below.

```
cat('The testing MAE for each of the folds in the 10-fold validation for  model m_sr are: \n',mae_m_sr =
```

```
## The testing MAE for each of the folds in the 10-fold validation for  model m_sr are:
##  0.4448058 0.2097074 0.2761283 0.3096626 0.2543357 0.3099679 0.5009914 0.231068 0.23142 0.2006918 0.1
```

The testing set mean absolute error (MAE) across the 10 folds in the k-fold cross validation for model m_sr is generated below.

```
cat('The testing MAE for the 10-fold validation for model m_sr is ', mean(MAE_m_sr), '.\n')
```

```
## The testing MAE for the 10-fold validation for model m_sr is  0.2968779 .
```

```
set.seed(1)
train = sample (200,200)
MSE = c()
MAE = c()
mse = c()
mae = c()

for (i in 1:10){
     train_data <- adv[-((i*20-19):(i*20)), ]
     test_data <- adv[((i*20-19):(i*20)), ]
     #m2_sr <- gam(Sales ~ s(TV) + s(Radio) + ti(TV, Radio), data = train_data)
     m2_sr <- gam(Sales ~ s(TV, k = 20) + s(Radio) + ti(TV, Radio), data = train_data)
     mse <- c(mse, mean((predict(m2_sr, newdata = test_data) - test_data$Sales)^2))

     test_pre <- predict(m2_sr, newdata = test_data)
     mae <- c(mae, mean(abs(test_pre - test_data$Sales)))

}
MSE_m2_sr = c(MSE, mean(mse))
MAE_m2_sr = c(MAE, mean(mae))
MSE_m2_sr
```

K-fold cross validation for model m2_sr and MAE output

```
## [1] 0.1763718
```

```
MAE_m2_sr
```

```
## [1] 0.2872649
```

```
summary(MAE_m2_sr)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2873  0.2873  0.2873  0.2873  0.2873  0.2873
```

The mean absolute errors (MAE) for the test set for each of the folds in the k-fold cross validation for model m_sr are generated below.

```
cat('The testing MAE for each of the folds in the cross-fold validation for model m2_sr are: \n',mae_m2
```

```
## The testing MAE for each of the folds in the cross-fold validation for model m2_sr are:
##  0.4270291 0.2421497 0.2767136 0.337132 0.2656718 0.3087673 0.4628276 0.2060792 0.1934478 0.1528311 (
```

The testing set mean absolute error (MAE) across the 10 folds in the k-fold cross validation for model m_sr is generated below.

```
cat('The testing MAE for the cross-fold validation for model m2_sr is', mean(MAE_m2_sr), '.\n')
```

```
## The testing MAE for the cross-fold validation for model m2_sr is 0.2872649 .
```

Summary: The lower mean absolute error is generated by the model m2_sr.

Model m_sr Test Dataset MAE: 0.2968779

Model m2_sr Test Dataset MAE: 0.2872649