Cross-Validation and Bias-Variance

Nate Wells

Math 243: Stat Learning

October 14th, 2020

Outline

In today's class, we will. . .

- Discuss variability in error estimates
- Investigate methods of cross-validation (LOOCV and k-fold cv)
- Implement CV in R
- Investigate the Bias-Variance trade-off

Section 1

Cross Validation

Penguins!

The `penguins` data set from the `palmerpenguins` package collected by Dr. Kristen Gorman on several attributes of antarctic penguins:

Penguins!

The penguins data set from the palmerpenguins package collected by Dr. Kristen Gorman on several attributes of antarctic penguins:

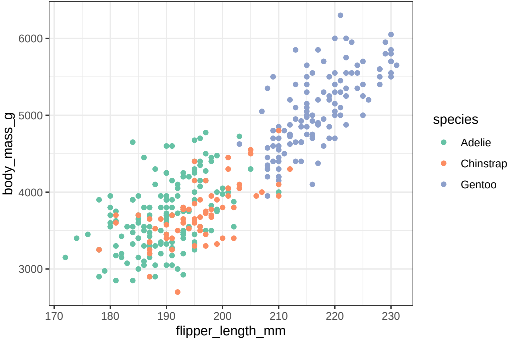
```
library(palmerpenguins)
data(penguins)
head(penguins)
```

```
## # A tibble: 6 x 8
##   species island bill_length_mm bill_depth_mm flipper_length_~ body_mass_g sex
##   <fct>   <fct>           <dbl>         <dbl>            <int>       <int> <fct>
## 1 Adelie  Torge~           39.1          18.7              181        3750 male
## 2 Adelie  Torge~           39.5          17.4              186        3800 fema~
## 3 Adelie  Torge~           40.3          18                195        3250 fema~
## 4 Adelie  Torge~           NA            NA                 NA          NA <NA>
## 5 Adelie  Torge~           36.7          19.3              193        3450 fema~
## 6 Adelie  Torge~           39.3          20.6              190        3650 male
## # ... with 1 more variable: year <int>
```

Penguin Species vs Body Mass and Flipper Length

Build LDA and QDA models for `species` as a function of `body_mass_g` and `flipper_length_mm`.

- Set a seed for reproducibility
- Use 70% of your data for training, and reserve remaining data for test.
- Record the error rates for each model on the google sheet on the course webpage



Example

```r
set.seed(1012)
penguins_train<-penguins %>% sample_frac(.7)
penguins_test<-penguins %>%  anti_join(penguins_train)

lda_mod<-lda(species ~ flipper_length_mm + body_mass_g, data = penguins_train)
qda_mod<-qda(species ~ flipper_length_mm + body_mass_g, data = penguins_train)

lda_pred<-predict(lda_mod, penguins_test)
qda_pred<-predict(qda_mod, penguins_test)
```

```r
lda_conf<-table(lda_pred$class, penguins_test$species)
lda_error<-(sum(lda_conf) - sum(diag(lda_conf)))/sum(lda_conf)

qda_conf<-table(qda_pred$class, penguins_test$species)
qda_error<-(sum(qda_conf) - sum(diag(qda_conf)))/sum(qda_conf)

data.frame(lda_error, qda_error)
```

```
##   lda_error qda_error
## 1 0.2673267 0.2376238
```

Leave One Out Cross Validation (LOOCV)

LOOCV partitions data into two sets:

- A single observation is randomly chosen as the test set
- All remaining observations are used as the training set to fit the model

Leave One Out Cross Validation (LOOCV)

LOOCV partitions data into two sets:

- A single observation is randomly chosen as the test set
- All remaining observations are used as the training set to fit the model

The process is repeated for each possible test set, and the average error rate computed among all partitions is computed

Leave One Out Cross Validation (LOOCV)

LOOCV partitions data into two sets:

- A single observation is randomly chosen as the test set
- All remaining observations are used as the training set to fit the model

The process is repeated for each possible test set, and the average error rate computed among all partitions is computed

The cross-validation estimate CV for average test MSE is therefore:

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{MSE}_i$$

Leave One Out Cross Validation (LOOCV)

LOOCV partitions data into two sets:

- A single observation is randomly chosen as the test set
- All remaining observations are used as the training set to fit the model

The process is repeated for each possible test set, and the average error rate computed among all partitions is computed

The cross-validation estimate CV for average test MSE is therefore:

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{MSE}_i$$

For the *very* special case of least squares regression, the following formula for CV holds:

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2 \qquad \text{where } h_i = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{j=1}^{n}(x_j - \bar{x})^2}$$

LOOCV in R

The `cv.glm()` function in the `boot` package can be used to perform LOOCV, and functions nearly identically to `glm()`.

```
library(boot)

penguins_nona <-penguins %>% drop_na()

my_glm<-glm(body_mass_g ~ flipper_length_mm , data = penguins_nona )
cv_error<- cv.glm(penguins_nona, my_glm)
cv_error$delta
```

```
## [1] 155563.6 155560.9
```

LOOCV in R

The `cv.glm()` function in the `boot` package can be used to perform LOOCV, and functions nearly identically to `glm()`.

```
library(boot)

penguins_nona <-penguins %>% drop_na()

my_glm<-glm(body_mass_g ~ flipper_length_mm , data = penguins_nona )
cv_error<- cv.glm(penguins_nona, my_glm)
cv_error$delta
```

```
## [1] 155563.6 155560.9
```

- The `cv.glm` function produces a list with several components. For now, we are interesetd in the estimate for test MSE, which is the first entry of the `delta` attribute.

LOOCV in R

The cv.glm() function in the boot package can be used to perform LOOCV, and functions nearly identically to glm().

```
library(boot)

penguins_nona <-penguins %>% drop_na()

my_glm<-glm(body_mass_g ~ flipper_length_mm , data = penguins_nona )
cv_error<- cv.glm(penguins_nona, my_glm)
cv_error$delta
```

```
## [1] 155563.6 155560.9
```

- The cv.glm function produces a list with several components. For now, we are interesetd in the estimate for test MSE, which is the first entry of the delta attribute.
- Our CV RSE is the square root of the CV MSE:

$$\text{CV RSE}_{(n)} = \sqrt{MSE} = \sqrt{155563.6} = 394.4$$

k-fold Cross Validation

What are some downsides of LOOCV?

k-fold Cross Validation

What are some downsides of LOOCV?

- Modeling fitting can be expensive, and LOOCV requires n fits. Most are very similar.

k-fold Cross Validation

What are some downsides of LOOCV?

- Modeling fitting can be expensive, and LOOCV requires n fits. Most are very similar.
- LOOCV may have high variance in estimating MSE, since the error outputs on each of the n models are highly correlated.

k-fold Cross Validation

What are some downsides of LOOCV?

- Modeling fitting can be expensive, and LOOCV requires n fits. Most are very similar.
- LOOCV may have high variance in estimating MSE, since the error outputs on each of the n models are highly correlated.

Let's cut back and instead fit only k models.

k-fold Cross Validation

What are some downsides of LOOCV?

- Modeling fitting can be expensive, and LOOCV requires n fits. Most are very similar.
- LOOCV may have high variance in estimating MSE, since the error outputs on each of the n models are highly correlated.

Let's cut back and instead fit only k models.

- Partition data into k sets of size n/k.
- Choose one subset of size n/k to be the test set, and remaining $k-1$ subsets to be training sets.
- Repeat for each subset of size n/k.

k-fold Cross Validation

What are some downsides of LOOCV?

- Modeling fitting can be expensive, and LOOCV requires n fits. Most are very similar.
- LOOCV may have high variance in estimating MSE, since the error outputs on each of the n models are highly correlated.

Let's cut back and instead fit only k models.

- Partition data into k sets of size n/k.
- Choose one subset of size n/k to be the test set, and remaining $k-1$ subsets to be training sets.
- Repeat for each subset of size n/k.

The k-fold CV method will give an intermediate estiamte for MSE between LOOCV and pure validation.

k-fold Cross Validation

What are some downsides of LOOCV?

- Modeling fitting can be expensive, and LOOCV requires n fits. Most are very similar.
- LOOCV may have high variance in estimating MSE, since the error outputs on each of the n models are highly correlated.

Let's cut back and instead fit only k models.

- Partition data into k sets of size n/k.
- Choose one subset of size n/k to be the test set, and remaining $k-1$ subsets to be training sets.
- Repeat for each subset of size n/k.

The k-fold CV method will give an intermediate estiamte for MSE between LOOCV and pure validation.

- Note that LOOCV is a special case of k-fold CV when $k = n$.

k-fold CV in R

We again use the cv.glm() function in the boot package.

```
library(boot)
set.seed(1)

penguins_nona <-penguins %>% drop_na()

my_glm<-glm(body_mass_g ~ flipper_length_mm , data = penguins_nona  )
cv_error<- cv.glm(penguins_nona, my_glm, K = 10)
cv_error$delta
```

```
## [1] 155065.5 154997.3
```

k-fold CV in R

We again use the `cv.glm()` function in the `boot` package.

```
library(boot)
set.seed(1)

penguins_nona <-penguins %>% drop_na()

my_glm<-glm(body_mass_g ~ flipper_length_mm , data = penguins_nona  )
cv_error<- cv.glm(penguins_nona, my_glm, K = 10)
cv_error$delta
```

```
## [1] 155065.5 154997.3
```

- This gives a 10-fold CV RSE of

$$\mathrm{CV\,RSE}_{(10)} = \sqrt{CV}_{(10)} = \sqrt{155065.5} = 393.8$$

Cross Validation for LDA and QDA

Both `lda` and `qda` have a LOOCV argument in their respective functions:

```
my_lda<-lda(species ~ flipper_length_mm , data = penguins_nona  , CV = T)
```

Cross Validation for LDA and QDA

Both `lda` and `qda` have a LOOCV argument in their respective functions:

```
my_lda<-lda(species ~ flipper_length_mm , data = penguins_nona  , CV = T)
```

When `CV = T`, both `lda` and `qda` return a list of `class` (the prediction) and `posterior` (the probabilitiy), derived from the LOOCV analysis.

Cross Validation for LDA and QDA

Both lda and qda have a LOOCV argument in their respective functions:

```
my_lda<-lda(species ~ flipper_length_mm , data = penguins_nona  , CV = T)
```

When CV = T, both lda and qda return a list of class (the prediction) and posterior (the probabilitiy), derived from the LOOCV analysis.

- No CV error rates are reported. But can be computed using table

Cross Validation for LDA and QDA

Both lda and qda have a LOOCV argument in their respective functions:

```
my_lda<-lda(species ~ flipper_length_mm , data = penguins_nona  , CV = T)
```

When CV = T, both lda and qda return a list of class (the prediction) and posterior (the probabilitiy), derived from the LOOCV analysis.

- No CV error rates are reported. But can be computed using table

```
head(my_lda$class)
```

```
## [1] Adelie Adelie Adelie Adelie Adelie Adelie
## Levels: Adelie Chinstrap Gentoo
```

```
head(my_lda$posterior)
```

```
##      Adelie  Chinstrap       Gentoo
## 1 0.9082461 0.09175317 7.235707e-07
## 2 0.8394852 0.16050009 1.475275e-05
## 3 0.6206141 0.37678880 2.597112e-03
## 4 0.6803584 0.31879476 8.468137e-04
## 5 0.7582377 0.24160938 1.528760e-04
## 6 0.9082461 0.09175317 7.235707e-07
```

Section 2

The Bias-Variance Trade-off

Example

See .html and .Rmd file on course webpage for live-coded notes