APTS Statistical Modelling: Practical 1

Helen Ogden

Suppose

$$y_{im} \sim \text{Poisson}\left(\mu(x_{im})\right),$$

independently, for $i = 1, \ldots, n$ and $m = 1, \ldots, M$, where

$$\mu(x_{im}) = 8 \exp\left(w(x_{im})\right),$$

for some function $w(.)$.

Suppose $M = 3$,

$$x_{im} = x_i = -10 + 20\,\frac{i-1}{n-1},$$

and

$$w(x) = 0.001\left(100 + x + x^2 + x^3\right).$$

Consider the following simulation study. For $b = 1, \ldots, B$:

- For $i = 1, \ldots, n$ and $m = 1, \ldots, M$, generate

  $$y_{im} \sim \text{Poisson}(\mu(x_{im})).$$

- Record the AIC for models

  $$y_{im} \sim \text{Poisson}(\mu(x_{im})), \qquad \mu(x_{im}) = \exp\left(\sum_{j=1}^{p} \beta_j x_{im}^{j-1}\right),$$

  for $p = 1, \ldots, p_{\max}$, where $p_{\max} = 20$.

You can run this simulation study with the following code:

```
B <- 1000
n <- 1000
M <- 3
pmax <- 20


w <- function(x) {
    0.001 * (100 + x + x^2 + x^3)
}
```

```r
mu <- function(x) {
    8 * exp(w(x))
}

x <- rep(seq(from = -10, to = 10, length = n), each = M)

aics <- matrix(0, nrow = B, ncol = pmax)

for(b in 1:B){
    y <- rpois(n = M * n, lambda = mu(x))

    mod <- glm(y ~ 1, family = poisson)
    aics[b, 1] <- AIC(mod)

    for(p in 2:pmax) {
        modp <- glm(y ~ poly(x, p - 1), family = poisson)
        aics[b,p] <- AIC(modp)
    }
}

AICorder <- apply(aics, 1, which.min) - 1
tAIC <- table(AICorder)
tAIC
```

Tasks

1. Modify the code above to investigate the performance of AIC as a model selection tool for $n = 25, 50, 100, 1000$. If your simulation study is taking too long to run, try reducing B to 100.
2. Vary the simulation model, using

$$w(x) = \frac{1.2}{1 + \exp(-x)},$$

   to see how AIC performs when the fitted models do not include the simulation model.
3. Modify the code to compute the values of BIC. Repeat the simulation studies from parts 1 and 2, using BIC to compare models. How do the results with AIC and BIC compare?

Solutions

We may put the code from the simulation study into a general function to allow us to vary n, M, $p_{\max}$, B, the function $w(.)$ and the information criteria used.

```
runsim <- function(n, M = 3, pmax = 20, B = 1000,
                   w =  function(x){0.001 * (100 + x + x^2 + x^3)},
                   crit = AIC) {
    mu <- function(x) {
        8 * exp(w(x))
    }

    x <- rep(seq(from = -10, to = 10, length = n), each = M)

    ics <- matrix(0, nrow = B, ncol = pmax)

    for(b in 1:B){
        y <- rpois(n = M * n, lambda = mu(x))

        mod <- glm(y ~ 1, family = poisson)
        ics[b, 1] <- crit(mod)

        for(p in 2:pmax) {
            modp <- glm(y ~ poly(x, p - 1), family = poisson)
            ics[b,p] <- crit(modp)
        }
    }

    ICorder <- apply(ics, 1, which.min) - 1
    table(ICorder)
}
```

1. ```
   runsim(n = 25)
   ```

   ```
   ## ICorder
   ##   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
   ## 729  87  62  32  22  10  11  10   9   8   6   1   3   4   3   1   2
   ```

   ```
   runsim(n = 50)
   ```

   ```
   ## ICorder
   ##   3   4   5   6   7   8   9  10  11  12  13  14  15  16  18
   ## 732 112  42  38  26  13   6   8   7   4   4   2   2   3   1
   ```

   ```
   runsim(n = 100)
   ```

   ```
   ## ICorder
   ```
   ```

```
##   3   4   5   6   7   8   9  10  11  12  13  14  15
## 734 110  43  35  30  12   7   5   7   7   4   3   3
```

```r
runsim(n = 1000)
```

```
## ICorder
##   3   4   5   6   7   8   9  10  11  12  13  14  15  18
## 730  89  51  47  21  12  18   7  11   4   3   2   4   1
```

The behaviour is similar for different n. In all cases, the correct (cubic) model is preferred most of the time, but the probability of it being selected does not tend to one as $n \to \infty$.

2.
```r
w2 <- function(x) {
    1.2 / (1 + exp(-x))
}
runsim(n = 25, w = w2)
```

```
## ICorder
##   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
##  90  34 337  84 204  61  72  34  27  11  15   9   5   8   4   4   1
```

```r
runsim(n = 50, w = w2)
```

```
## ICorder
##   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
##   6   2 220  68 325  82 132  45  49  16  18  11   3   8   6   5   4
```

```r
runsim(n = 100, w = w2)
```

```
## ICorder
##   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
##  65  23 327  85 224  59  87  46  18  17  16  10   7   7   9
```

```r
runsim(n = 1000, w = w2)
```

```
## ICorder
##   9  10  11  12  13  14  15  16  17  18  19
## 107  45 374  83 192  43  76  29  23  11  17
```

As n increases, AIC tends to select increasingly complex models, which provide a better approximation to the true distribution which generated the data, which is not a polynomial model.

3. We can redo all calculations for both cases of the function $w(.)$ for BIC. For the case where the cubic model is correct:

```r
runsim(n = 25, crit = BIC)
```

```
## ICorder
```

```
##    1    2    3    4    5    6    7
##    8    4  950   28    5    4    1
```

```r
runsim(n = 50, crit = BIC)
```

```
## ICorder
##   3   4
## 975  25
```

```r
runsim(n = 100, crit = BIC)
```

```
## ICorder
##   3   4   5
## 979  19   2
```

```r
runsim(n = 1000, crit = BIC)
```

```
## ICorder
##   3   4
## 996   4
```

As n increases, the probability that BIC selects the correct (cubic) model tends to 1.

For the case `w = w2`, where none of the models are correct:

```r
runsim(n = 25, w = w2, crit = BIC)
```

```
## ICorder
##    1    3    4    5    6    7    8    9   10
##    2  405   48  419   39   70    7    9    1
```

```r
runsim(n = 50, w = w2, crit = BIC)
```

```
## ICorder
##   3   4   5   6   7   8   9
## 163  23 598  53 134  16  13
```

```r
runsim(n = 100, w = w2, crit = BIC)
```

```
## ICorder
##   3   4   5   6   7   8   9  11
##  11   2 580  54 311  15  25   2
```

```r
runsim(n = 1000, w = w2, crit = BIC)
```

```
## ICorder
##   7   8   9  10  11  12  13
## 164  19 702  28  85   1   1
```

BIC prefers simpler models to AIC, although it still tends to prefer more complex models as n increases in this case.