Forming Multiple Confidence Intervals for Model Selection

Norm Matloff

Table of contents

Author bio

In developing/choosing a model, we may perform simulation or cross-validation, with many replicates, in order to compare multiple algorithms, multiple sets of hyperparameters and so on. To make our analysis statistically valid, we can form confidence intervals (CIs).

However, if we form many CIs, say at 95% level each, their overall coverage probability will be much lower than 95%. This is the *multiple inference* (MI) or *multiple comparisons* problem, sometimes also called *simultaneous inference*. In this document, we discuss remedies in the model-selection context. (General application of MI methods will be presented as well.)

The MI problem has been extremely well studied, resulting in myriad methods. Here we employ two of the most well-known methods, the Bonferroni Inequality and Scheffe's Method.

For a good general presentation of MI methods, see Jason Hsu, *Multiple Comparisons: Theory and methods.*

Note that our focus is on CIs, not hypothesis tests. We strongly recommend against using the latter in any statistical analysis.

Motivating Example

```
library(qeML)
data(svcensus)
head(svcensus)
```

This is US census data. Let's predict gender, using logistic regression, random forests, XGBoost, and k-NN with 2 values of k..

```
logitAcc <- qeLogit(svcensus,'gender')$testAcc
rfAcc <- qeRFranger(svcensus,'gender')$testAcc
xgbAcc <- qeXGBoost(svcensus,'gender')$testAcc
knn25Acc <- qeKNN(svcensus,'gender',k=25)$testAcc
knn200Acc <- qeKNN(svcensus,'gender',k=200)$testAcc
c(logitAcc,rfAcc,xgbAcc,knn25Acc,knn200Acc)
# 0.263 0.254 0.263 0.256 0.236
```

Several points to note:

- The **qeML** functions automatically do cross-validation, via an argument **holdout**, indicating our desired size of test set. Here we take the default value.

- The functions return S3 objects, one of whose components is prediction accuracy on the test data, **testAcc**, in this case the probability of misclassification..
- Since the holdout set is random, we should be performing each of the calls many times, and compute averages, say

```
logitAccs <-
   replicate(50,qeLogit(svcensus,'gender')$testAcc)
rfAccs <-
   replicate(50,qeRFranger(svcensus,'gender')$testAcc)
xgbAccs <-
   replicate(50,qeXGBoost(svcensus,'gender')$testAcc)
knn25Accs <- replicate(50,qeKNN(svcensus,'gender',k=25)$testAcc)
knn200Accs <- replicate(50,qeKNN(svcensus,'gender',k=200)$testAcc)
accs <- cbind(logitAccs,rfAccs,xgbAccs,knn25Accs,knn200Accs)
colMeans(accs)
# 0.24476    0.26124    0.25828    0.25194    0.24844
```

We might wish to find a CI for each of the 5 quantities, or for each difference, i.e. c(5,2) = 10 CIs. With more algorithms, and especially with more hyperparameter combinations, our CI count could easily be several dozen or more, raising MI concerns.

Review: Confidence Intervals, Standard Errors

To set the stage, let's review the statistical concepts of *confidence interval* and *standard error*. Say we have an estimator $\hat{\theta}$ of some population parameter θ, e.g.
$\bar{X}$ for a population mean μ.

We follow the standard model in statistics in which one's data are considered a sample from some population, actual or conceptual. In the ML community, the term *data-generating process* is analogous. Also note the "hat" notation, '$\widehat{\ }$' meaning "estimate of."

Loosely speaking, the term *standard error* of $\hat{\theta}$ is our estimate of $\sqrt{Var(\hat{\theta})}$. More precisely, suppose that $\hat{\theta}$ is asymptotically normal/Gaussian. The standard error is an estimate of the standard deviation of that normal distribution. For this reason, one sometimes writes $AVar(\hat{\theta})$ rather than $Var(\hat{\theta})$.

In the familiar case in which θ is a population mean and $\hat{\theta}$ is the sample mean,

$$s.e.(\widehat{\theta}) = \frac{s}{\sqrt{n}} \tag{1}$$

where s^2 is the sample variance. and where we denote the standard error of $\widehat{\theta}$ by s.e.$(\widehat{\theta})$.

An approximate 95% confidence interval (CI) for θ is then

Standard statistics courses begin with "exact" CIs, using the t-distribution, which assumes X has a normal dstr. But nothing in practice is normally distributed; no one is 80 feet tall, for instance

$$\widehat{\theta} \pm 1.96 \text{ s.e.}(\widehat{\theta})$$

The 95% figure means that of all possible samples of the given size from the population, 95% of the resulting confidence intervals will contain θ. In many cases, the 95% figure is only approximate, stemming from a derivation that uses the Central Limit Theorem.

In general, for confidence level $1 - \alpha$, replace 1.96 by z_α, the $1 - \alpha/2$ quantile of the N(0,1) distribution, Then our CI is

$$\widehat{\theta} \pm z_\alpha \text{s.e.}(\widehat{\theta}) \tag{2}$$

Examples of finding z_α:

```
> qnorm(0.975)
[1] 1.959964  # for 95% CI
> qnorm(0.025)  # N(0,1) is symmetric around 0
[1] -1.959964
> qnorm(0.995)  # for 99% CI
[1] 2.575829
```

Example: Logistic regression coefficients

```
suppressPackageStartupMessages(library(qeML))
data(svcensus)
logitOut <- qeLogit(svcensus,'gender',yesYVal='female')
summary(logitOut$glmOuts[[1]])
```

```
Call:
glm(formula = yDumm ~ ., family = binomial, data = tmpDF)

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)  -5.635e-01  9.435e-02  -5.972 2.34e-09 ***
age           5.060e-03  1.543e-03   3.279 0.001043 **
educ16       -5.458e-01  1.178e-01  -4.634 3.60e-06 ***
educzzzOther -8.017e-02  4.359e-02  -1.839 0.065867 .
occ101       -3.635e-01  4.789e-02  -7.590 3.20e-14 ***
occ102       -3.661e-01  4.484e-02  -8.166 3.19e-16 ***
occ106        3.650e-01  9.854e-02   3.704 0.000212 ***
occ140       -8.998e-01  1.052e-01  -8.550  < 2e-16 ***
occ141       -1.461e+00  7.327e-02 -19.942  < 2e-16 ***
wageinc      -6.181e-06  5.237e-07 -11.802  < 2e-16 ***
wkswrkd       9.519e-04  1.281e-03   0.743 0.457515
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 21240  on 19089  degrees of freedom
Residual deviance: 20339  on 19079  degrees of freedom
AIC: 20361

Number of Fisher Scoring iterations: 4
```

So a 95% CI for the coefficient for occupation 141 is

$$-1.44 \pm 1.96 \times 0.07 \tag{3}$$

The **qeLogit** function is a wrapper for **glm**. In order to accommodate more than two classes, it performs multiple **glm** runs storing their outputs in an R list **glmOuts**.

The Bonferroni Inequality

This one is the simplest and most convenient MI method. The derivation is instructive.

Suppose A and B are events defined on some probability space. Then

$$P(A \text{ or } B) \leq P(A) + P(B) \tag{4}$$

Say we form 95% CIs for two different quantities, so that each has a probability 5% of being "wrong," i.e. of failing to contain its corresponding population parameter. Set A to be the event that the first CI fails in that regard, and define B similarly for the second CI. Then Equation 4 tells us that the probability of at least one of the CIs being wrong is at most 10%. In other words, *our overall confidence level is at least 90%.*

Moreover: Presumably the reason we set the original CI levels to 95% was that we are comfortable with an error rate of 5%. Accordingly, we may wish to have an *overall* rate of 5% – again, meaning that we are 95% confident that *both* CIs are correct – rather than the 90% shown above. The same reasoning as above shows that we can achieve overall 95% confidence by making each of the two individual CIs at the 97.5% level.

So we could form intervals for say both **occ140** and **occ141** above.

Of course, this MI benefit comes at a price:

```r
qnorm((1-0.975)/2)
# -2.241403
```

So, the 1.96 in Equation 2 now becomes 2.24, forcing us to form wider intervals.

If in Equation 4 one replaces B by B_1 or B_2, that extends the relation from two events to three, and so on.

For events $A_1, ..., A_m$,

$$P(A_1 \text{ or } ... \text{ or } A_m) \leq \sum_{i=1}^{m} P(A_i)$$

Scheffe's Method, Background Prep

The Bonferroni Method is fine for forming a few CIs, but is impractical if one needs many. The coefficient for forming a CI – 1.96 and 2.24 above – becomes too large. By contrast, the Scheffe' Method actually gives us the freedom to form infinitely many CIs.

Definitions

The term *covariance* is overloaded, with both scalar and matrix versions.

- The covariance between random variables U and V is

$$Cov(U,V) = E[(U-EU)(V-EV)]$$

- The *covariance matrix* of a random vector X has as its row i, column j element the scalar covariance between elements i and j (elements denoted here by superscipts):

$$Cov(X)_{ij} = Cov(X^{(i)}, X^{(j)})$$

  One can show that

$$Cov(X) = E[(X-EX)(X-EX)'] \tag{5}$$

  where ' denotes matrix transpose and vectors are column vectors by default. Here EX is a vector whose element i is $EX^{(i)}$.

Estimation via sample analog (“plug-In” estimates)

Denote our data by $X_1, ..., X_n$. Equation 5 is the average value of $(X - EX)(X - EX)'$ in the population, and its sample analog is the average value in the sample, i.e.

$$\widehat{Cov}(X) = \frac{1}{n} \sum_{i=1}^{n} (X_i - \bar{X})(X_i - \bar{X})' \tag{6}$$

where

$$\bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

In R, say we have a data frame or matrix **A**, one data point per row. Then **cov(A)** computes Equation 6.

Note too that in analogy with Equation 1,

$$\widehat{Cov}(\bar{X}) = \frac{1}{n} \widehat{Cov}(X)$$

Estimation via parametric model: the R vcov function

Parametric statistical models, e.g. linear and logistic regression, parametric analysis of contingency tables and so on, are smooth functions of sums, and thus asymptotically normally distributed. Their covariance matrices arise as part of the derivation, and are accessible for many R estimators via the **vcov** function, such as in our logit model above:

The **vcov** function is an R *generic* function, playing a similar role to **print**, **plot**, **summary** and so on. Many R statistical operations have generic functions available to apply to their output, including say **glm**. When we make the call **vcov(glmOut)**, the R interpreter sees that **glmOut** is of class “glm” and thus transfers the call to the class-specific function, **vcov.glm(glmOut)**.

```
vcov(logitOut$glmOuts[[1]])
```

```
               (Intercept)           age       educ16  educzzzOther
(Intercept)   8.901692e-03 -9.775752e-05 -1.111155e-03 -1.731084e-03
age          -9.775752e-05  2.381474e-06 -7.749125e-06  1.871031e-06
educ16       -1.111155e-03 -7.749125e-06  1.387431e-02  1.410175e-03
educzzzOther -1.731084e-03  1.871031e-06  1.410175e-03  1.899972e-03
occ101       -1.254284e-03  3.187716e-06  1.067054e-04  2.579301e-05
```

```
occ102       -1.446238e-03  6.654231e-06  7.911654e-05  2.291348e-04
occ106       -1.108154e-03  3.163325e-06  9.248541e-05  6.536398e-05
occ140       -1.337016e-03  5.544354e-06 -8.487479e-05  1.389349e-04
occ141       -8.735317e-04 -8.070922e-06  1.603566e-05  1.842955e-04
wageinc       3.526441e-09 -1.062608e-10 -4.916852e-10  3.208639e-09
wkswrkd      -5.998181e-05  1.270412e-07 -5.947216e-07 -2.449374e-06
                    occ101        occ102        occ106        occ140
(Intercept)  -1.254284e-03 -1.446238e-03 -1.108154e-03 -1.337016e-03
age           3.187716e-06  6.654231e-06  3.163325e-06  5.544354e-06
educ16        1.067054e-04  7.911654e-05  9.248541e-05 -8.487479e-05
educzzzOther  2.579301e-05  2.291348e-04  6.536398e-05  1.389349e-04
occ101        2.293429e-03  1.058975e-03  1.045571e-03  1.052902e-03
occ102        1.058975e-03  2.010277e-03  1.073871e-03  1.110842e-03
occ106        1.045571e-03  1.073871e-03  9.710788e-03  1.063072e-03
occ140        1.052902e-03  1.110842e-03  1.063072e-03  1.107483e-02
occ141        1.038412e-03  1.090634e-03  1.047558e-03  1.068253e-03
wageinc      -7.285088e-10 -3.738745e-09 -8.642752e-10 -2.361078e-09
wkswrkd       2.156195e-06  2.614042e-06 -1.767986e-06  1.509317e-06
                    occ141       wageinc       wkswrkd
(Intercept)  -8.735317e-04  3.526441e-09 -5.998181e-05
age          -8.070922e-06 -1.062608e-10  1.270412e-07
educ16        1.603566e-05 -4.916852e-10 -5.947216e-07
educzzzOther  1.842955e-04  3.208639e-09 -2.449374e-06
occ101        1.038412e-03 -7.285088e-10  2.156195e-06
occ102        1.090634e-03 -3.738745e-09  2.614042e-06
occ106        1.047558e-03 -8.642752e-10 -1.767986e-06
occ140        1.068253e-03 -2.361078e-09  1.509317e-06
occ141        5.369049e-03 -2.536243e-09  2.676902e-06
wageinc      -2.536243e-09  2.742458e-13 -3.285020e-10
wkswrkd       2.676902e-06 -3.285020e-10  1.641559e-06
```

Keep in mind that our θ here is β, the vector of regression coefficients. So, the covariance matrix printed out here is the estimated asymptotic covariance matrix of $\widehat{\beta}$. So for instance the estimated covariance between $\widehat{\beta}_{occ101}$ and $\widehat{\beta}_0$ is about -0.00127.

The multivariate normal distribution family

This is an extension of the ordinary normal family, with parameters mean vector and covariance matrix. Again, “there is no such animal” in practice, but fortunately there is a multivariate version of the Central Limit Theorem, which makes it all work.

The chi-square family of probability distributions

Some readers may have seen this distribution family in the context of goodness-of-fit tests. Here is the technical definition:

The random variable W is said to have a *chi-square distribution* with k *degrees of freedom* if W has the same distribution as

$$Z_1^2 + ... + Z_k^2$$

for independent Z_i having N(0,1) distributions.

So, unlike the 2-parameter family of normal distributions, chi-square has just 1 parameter, the degrees of freedom. Note too that it is not symmetric.

Key theorem

Say the random vector X of length p is multivariate normal, with mean vector ν and (invertible) covariance matrix Λ. Then the quantity

$$(X-\nu)'\Lambda^{-1}(X-\nu)$$

has a chi-square distribution with p degrees of freedom.

If ν and Λ are estimated from the sample as above, the distribution is approximate.

Scheffe' CIs

> Say $\widehat{\theta}$ is an asymptotically k-dimensional normal estimator of some population value θ. Denote the covariance matrix in that asymptotic distribution by Σ, assumed invertible, with estimate $\widehat{\Sigma}$. Denote the sample size by n. Then for any constant, compatible vector a, form the interval
>
> $$a'\widehat{\theta} \pm \sqrt{d_\alpha a'\widehat{\Sigma}a} = a'\widehat{\theta} \pm \sqrt{d_\alpha}\ s.e.(a'\widehat{\theta}) \qquad (7)$$
>
> where d_α is the upper α quantile of the chi-square distribution with k degrees of freedom.
>
> Then the intervals in Equation 7 hold simultaneously at the approximate $(1-\alpha)$ confidence level.

The values of a are chosen according to the analyst's interests. For instance, in our logit model above, we could take $a = (0,0,0,0,1,0,...,0)$ if we want a CI for β_{occ101}, or $a = (0,0,0,0,1,-1,0,...,0)$ if we want a CI for the difference $\beta_{occ101} - \beta_{occ102}$. In any event, the key point is that we will be forming at least several of these CIs, if not many.

In the case $a = (0,0,0,0,1,-1,0,...,0)$, the sum of the elements of a is 0, known as a *contrast*. If one is willing to restrict one's analysis to contrasts, the degrees of freedom will decline by 1, making for slightly narrower CIs.

Example

Let's continue the example in Section .

```
logitAccs <-
   replicate(50,qeLogit(svcensus,'gender')$testAcc)
rfAccs <-
   replicate(50,qeRFranger(svcensus,'gender')$testAcc)
xgbAccs <-
   replicate(50,qeXGBoost(svcensus,'gender')$testAcc)
knn25Accs <- replicate(50,qeKNN(svcensus,'gender',k=25)$testAcc)
knn200Accs <- replicate(50,qeKNN(svcensus,'gender',k=200)$testAcc)
accs <- cbind(logitAccs,rfAccs,xgbAccs,knn25Accs,knn200Accs)
covaccs <- cov(accs)
xbar <- colMeans(accs)
covxbar <- covaccs / nrow(accs)
```

```
d <- sqrt(qchisq(0.95,5))  # 5 degrees of freedom
# form CI for difference between logit and RF
a <- c(1,-1,0,0,0)
t(a) %*% xbar
# -0.01258
d * sqrt(t(a) %*% covxbar %*% a)  # radius of CI
t(a) %*% xbar - d * sqrt(t(a) %*% covxbar %*% a)
t(a) %*% xbar + d * sqrt(t(a) %*% covxbar %*% a)
# CI is (-0.02174126,-0.003418737)
```

Again, note that we can form as many CIs this way as we want, yet still retain an overall confidence level of at least 95%.

Impact of the Sampling Scheme

Again consider the code in Section . Since the random number seed is not reset, each of the calls is using different training sets and different test sets from each other. This makes them statistically independent. The alternative would be to insert, say

```
set.seed(9999)
```

before each of the calls:

```
set.seed(9999)
logitAccs <-
   replicate(50,qeLogit(svcensus,'gender')$testAcc)
set.seed(9999)
rfAccs <-
   replicate(50,qeRFranger(svcensus,'gender')$testAcc)
set.seed(9999)
xgbAccs <-
   replicate(50,qeXGBoost(svcensus,'gender')$testAcc)
set.seed(9999)
knn25Accs <- replicate(50,qeKNN(svcensus,'gender',k=25)$testAcc)
set.seed(9999)
knn200Accs <- replicate(50,qeKNN(svcensus,'gender',k=200)$testAcc)
accs1 <- cbind(logitAccs,rfAccs,xgbAccs,knn25Accs,knn200Accs)
```

Now each of the 5 fits will be to the same dataset.

Let's call these two sampling schemes the Independent Scheme and the Same Dataset Scheme.

Now, what are the pros and cons here?

Under the Same Dataset Scheme, the fact that each call operates on the same dataset seems appealing – say on the grounds of "an apples to apples comparison." And that intuition is supported by a closer look at the math, as follows.

Recall that

$$Var(V - U) = Var(V) + Var(U) - 2Cov(U, V) \tag{8}$$

Now apply that to our example above in which we compared the random forest solution to logistic regression, setting

$$a < -c(1, -1, 0, 0, 0)$$

The key point is since in the second scenario the various algorithms are all applied to the same dataset, the results are positively correlated. Equation 8 then tells us that under this scheme, the standard error of the difference will be reduced, relative to the independent sampling scheme.

Let's take a look:

```
set.seed(9999)
logitAccs <-
   replicate(50,qeLogit(svcensus,'gender')$testAcc)
set.seed(9999)
rfAccs <-
   replicate(50,qeRFranger(svcensus,'gender')$testAcc)
set.seed(9999)
xgbAccs <-
   replicate(50,qeXGBoost(svcensus,'gender')$testAcc)
set.seed(9999)
knn25Accs <- replicate(50,qeKNN(svcensus,'gender',k=25)$testAcc)
set.seed(9999)
knn200Accs <- replicate(50,qeKNN(svcensus,'gender',k=200)$testAcc)
```

```
accs1 <- cbind(logitAccs,rfAccs,xgbAccs,knn25Accs,knn200Accs)
covaccs1 <- cov(accs1)
xbar1 <- colMeans(accs1)
covxbar1 <- covaccs1 / nrow(accs1)
t(a) %*% xbar  # -0.01258
t(a) %*% xbar1  # -0.0117
d * sqrt(t(a) %*% covxbar1 %*% a)  # 0.005706407
d * sqrt(t(a) %*% covxbar %*% a)  # 0.009161263
```

So use of the Same Dataset Scheme cut the radius of the CI by almost half!

Now, what about the Independent Scheme? Note first that the analysis in Section *is* valid, but there is an opportunity cost (though probability small) in not taking advantage of the independence. We could do the latter by setting the off-diagonal elements of the covariance matrix to 0s.

Notes

- Some analysts may not consider MI to be a "problem." This is a philosophical issue, not pursued here, but we note that the choice may depend on the application. A medical research journal may require MI, say, whereas an amateur stock market investor may not feel it's necessary for that type of data analysis.

- There is a bit of drama in this word *contain* in the statement "95% of the resulting confidence intervals will contain θ." Instead of saying the intervals *contain* θ, why not simply say θ is *in* the intervals? Aren't these two descriptions equivalent in terms of English?

  Of course they are. But many instructors of statistics classes worry that students will take the description based on "in" to mean that θ is the random quantity, when in fact the CI is random (random center, random radius) and θ is fixed (though unknown). The instructors thus insist on the more awkward phrasing "contain," so as to avoid students misunderstanding. Indeed some instructors would contend that use of the word *in* is itself just plain incorrect.

My own view is that in some cases the word *in* is clearer (and certainly correct in any case), and that it is better to add a warning about what is random/nonrandom than engage in awkward phrasing.