The Delta Method for Variance of Sample Statistics and hypothesis testing of micro-averaged plugin estimates.

Pushpendre Rastogi
January 25, 2021

The delta method is used to estimate the variance of (a function of a asymptotically normally distributed estimator, say E) in terms of the variance of E.

For example, say we want to estimate a function of two population parameters θ_1 and θ_2. We have collected n samples and we used those samples to compute the estimators $\hat{\theta}_1$ and $\hat{\theta}_2$. Also our procedure for computing $\hat{\theta}_1$ and $\hat{\theta}_2$ is consistent, in other words $\hat{\theta}_1, \hat{\theta}_2$ are consistent. And $\hat{\theta}_1, \hat{\theta}_2$ are jointly asymptotically normally distributed around the true values. And we know the variance of the estimators $(\hat{\theta}_1, \hat{\theta}_2)$. Then we can compute the variance of the plugin estimator $f(\hat{\theta}_1, \hat{\theta}_2)$

Even more concretely, say we want to estimate the "mean order value" over all customers in last 12 months. Our transaction data is collected in terms of single orders, which we'll need to group by customers and then aggregate. So for each customer we will observe a two dimensional random variable, the first dimension is the number of orders, and the second dimension is the order value summed over all orders. Now there are two ways of estimating the **population average order value**, the so-called *micro averaging* method and the second one which is the **macro-averaging** method.

1. **micro-average** First sum the total order value over all customers, then divide that by the total orders over all customers. This is akin to first estimating the **mean order value per customer** $\hat{\theta}_1$ and dividing it by **mean number of orders per customer** $\hat{\theta}_2$, and then computing the plugin estimator $f(\hat{\theta}_1, \hat{\theta}_2) = \hat{\theta}_1 / \hat{\theta}_2$.

2. **macro-average** For each customer first divide the total value by the total number of orders, and then take the average of that ratio. This approach first estimates a ratio for each customer – which can be very noisy – due to small large variation in the denominator and then averages the ratios.

Another example, is that say we have customers coming to us repeatedly in different sessions over a period of time. For each session we measure their engagement with our service. So for each customer we will observe a tuple of total sessions, and total engagement across all sessions, and then we want to estimate the mean engagement over all sessions. The same consideration of micro-vs-macro averaging will apply here as well.

Now the **Delta method** can be used to estimate the variance of the **micro-average** estimator, and the result is as follows.

Let $\hat{\boldsymbol{\theta}}$ denote a vector of estimators with asymptotic distribution $\mathcal{N}(\boldsymbol{\theta}, \boldsymbol{\Sigma})$, then the asymptotic distribution $f(\theta) = \mathcal{N}(f(\boldsymbol{\theta}), \nabla f(\boldsymbol{\theta})^T \ \boldsymbol{\Sigma} \ \nabla f(\boldsymbol{\theta})$

Finally in practice this estimate of the variance can be used to normalize the z-score for comparing two treatments via a z-test.

1 Simulation

Consider a population where the number of orders per customer is poisson distributed with maximum value of 100, and the value per order has the log-normal distribution with parameters having uniform prior. The variance in micro average method is 2/3 of the variance of the macro-average method. Therefore, there is benefit in using the delta method. Better code is given at `https://stats.stackexchange.com/questions/398436/`

```

library(ggplot2)
> den4 <- density(rgamma(10000, shape=4, rate=1))
> den22 <- density(rgamma(10000, shape=2, rate=1)
    + rgamma(10000, shape=2, rate=1))
> (ggplot(data=data.frame(x=den4$x, y=den4$y),
         aes(x=x, y=y))
   + geom_point()
   + geom_point(data=data.frame(x=den22$x, y=den22$y), color="red")
   + theme_classic())

# The difference between macro averaging and micro averaging
# does not matter, if we get enough counts per customer.
# i.e. k <- pmin(rpois(n, 10), 100)
# The difference between macro and micro also doesn't matter
# if the order scale is the same for each customer.
# i.e. v <- sapply(k, function(x) rgamma(1, 2 * x, rate=1))
# [1] 8.223577 8.230807 2.169754
# At a cursory glance it may seem that the delta method is
# overkill for the ratio metrics.
# but becomes much more significant when there are not a
# lot of purchases per customer, as is common in e-commerce.
# And we are doing A/B testing, where a 10\% reduction in
# standard deviation means a 10\% reduction in the cost
# of running an experiment.

trial1 <- function() {
 n <- 10000
 k <- pmin(rpois(n, 1), 100)
 v <- sapply(k, function(x) if(x==0) 0 else sum(sapply(1:x,
    function(xx) rlnorm(1, meanlog=runif(1, min=0.1, max=2), sdlog=
    runif(1, min=0.1, max=2)))))
 mean(v)/mean(k)
}
trial2 <- function() {
 n <- 10000
 k <- pmin(rpois(n, 1), 100)
 v <- sapply(k, function(x) if(x==0) 0 else sum(sapply(1:x,
    function(xx) rlnorm(1, meanlog=runif(1, min=0.1, max=2), sdlog=
    runif(1, min=0.1, max=2)))))
 mean(v[k > 0] / k[k > 0])
}
r1 <- sapply(1:100, function (x) trial1())
r2 <- sapply(1:100, function (x) trial2())
print(c(mean(r1), mean(r2), sqrt(var(r1)), sqrt(var(r2))))

[1] 8.0904948 8.1147581 0.4277769 0.5947738
```

2 Recall Estimation

Another possibility is to use this delta method for estimating the recall of a system. Let's say we are constructing a binary classifier $f : \mathcal{X} \rightarrow \{0, 1\} = \mathcal{Y}$. The precision of a system is $\pi = \mathbb{P}[\mathbb{I}[Y = 1] \mid \mathbb{I}[f(X) = 1]$, and the recall is $r = \mathbb{P}[\mathbb{I}[f(X) = 1] \mid \mathbb{I}[Y = 1]]$ therefore the recall is precision multiplied with $\frac{\mathbb{P}[f(X)=1]}{\mathbb{P}[Y=1]}$. This is exactly the problem of estimating a ratio. Note that $\mathbb{P}[f(X) = 1]$ can be estimated very easily on a very large unlabeled dataset.

If the base-rate $P[Y = 1]$ is 0.01 then annotating $10,000$ samples will result in 100 positive examples, and if our true recall is 0.5, true precision is 0.9, then the true value of the numerator will be $0.5/0.9 * 0.01 = 0.0055$.