Resampling based testing for linear regression

Bootstrap is extremely useful to compute standard deviation or stand error of an estimate, hypothesis testing and many more. We learn this by computing the standard error of the regression coefficient β for the following linear regression problem $y_i = \beta x_i + \epsilon_i$.

```
x <- rnorm(100)
y <- rnorm(100, 2*x)

#generate a dummy variable
z <- rnorm(100)
```

However, we will fit the following two models $y_i = x_i\beta_1 + \epsilon_i$ and $y_i = x_i\beta_1 + z_i\beta_2 + \epsilon_i$. Thus, the true values are $\beta_{10} = 2$ and $\beta_{20} = 0$.

Let $X = [x; z]$, concatenating the columns.

- Assuming asymptotic normality or normality of the error, $V(\hat{\beta}) = V\{(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}\} = (\mathbf{X}^T\mathbf{X})^{-1}\hat{\sigma}^2$, where $\hat{\sigma}^2 = \frac{1}{n-p}(\mathbf{y} - \mathbf{X}\hat{\beta})^2$. Here $p = 1$, for the first case and $p = 2$ for the second case. $[V(y) = \sigma^2]$
- Another way is just to apply `summary(lm(y~x-1))` or `summary(lm(y~X-1))`. Note that `lm` reports standard error, which is the standard deviation.

Both of the above two approaches will give you theoretical values. Now we compute the same using bootstrap samples on the first model and repeat the same steps for model 2 as well.

1. Define an array `betaboot <- rep(0, 1000)` and matrix `betaboot2 <- matrix(0, 1000, 2)`
2. Use R function `sample` to draw indices `ind` from 1:100 with replacement of size 100. Thus, some of the generated indices might repeat.
3. Apply `fit <- lm(y[ind]~ x[ind]-1)`. and Apply `fit2 <- lm(y[ind]~ X[ind,]-1)`.
4. Store `betaboot[1] <- fit$coefficient` and `betaboot2[1,] <- fit2$coefficient`
5. Repeat above steps 2:4 for 1000 times and store the coefficients one by one in `betaboot`.
6. In the end you have bootstrap samples of β
7. Calculate `var(betaboot)`, `var(betaboot2[,1])`, `var(betaboot2[,2])` and compare with their theoretical values.

If you apply `summary(lm(y x-1))`, you get a theoretical standard error of β. But now, you resample the data as before, estimate $\hat{\beta}$'s for different resamples applying `lm` or something similar on the resampled data. This will give you bootstrap samples of β. Use those to compute standard deviation and also bootstrap confidence interval. Compare this standard deviation with full data estimated standard error of `summary(lm(y x-1))`.

```
x <- rnorm(100)
y <- rnorm(100, 2*x)

summary(fit <- lm(y~x-1))

sigma2hat <- sum(fit$residuals^2)/(100-1)

betavar <- sigma2hat/sum(x*x)
betasd <- sqrt(betavar)
betasd
betap <- rep(0, 10000)

for( i in 1:10000){
  ind <- sample(1:100, 100, replace = T)

  fit <- lm(y[ind]~x[ind]-1)
  betap[i] <- fit$coefficient
}

sd(betap)

mean(betap)

#Empirical CI using bootstrap samples
quantile(betap, probs = c(0.025, 0.975))
```

Now, we make the problem a bit more interesting for illustration purpose and to show that these bootstrap samples can be used for hypothesis testing.

```
#Making the problem a bit more interesting
z <- rnorm(100) #y does not depend on this z.
```

```
X <- cbind(x, z)

summary(fit <- lm(y~X))

sigma2hat <- sum(fit$residuals^2)/(100-2)

betavar <- solve(crossprod(X))*sigma2hat
beta1sd <- sqrt(betavar[1,1])
beta2sd <- sqrt(betavar[2,2])
beta1sd
beta2sd

betap <- matrix(0, 10000, 2)

for( i in 1:10000){
  ind <- sample(1:100, 100, replace = T)

  fit <- lm(y[ind]~x[ind]+z[ind]-1)
  betap[i, ] <- fit$coefficient
}

apply(betap, 2, sd)
apply(betap, 2, mean)

#Empirical CI using bootstrap samples
apply(betap, 2, quantile, probs = c(0.025, 0.975))
```

Check that empirical bootstrap CI of `beta[2]` contain zero, hence insignificant.

Above 'Bootstrap' approach did not require any assumption on the distribution of beta. Hence, it is a non-parametric approach. There also exists parametric bootstrap. Parametric bootstrapping assumes that the data comes from a known distribution with unknown parameters. (For example, the data may come from a Poisson, negative binomial for counts, or normal for continuous distribution.) You estimate the parameters from the data that you have, and then you use the estimated distributions to simulate the samples. We see this example in HW.