Problem Set 2

Shuai Yuan

February 3, 2020

Question 1

A regression that contains all the variables in the dataset was run against the feeling towards Biden.

```r
#Model Fitting
linear <- lm(biden~ ., data = nes2008)
linear%>%
  tidy()%>%
  kable()
```

term	estimate	std.error	statistic	p.value
(Intercept)	58.8112590	3.1244366	18.822996	0.0000000
female	4.1032301	0.9482286	4.327258	0.0000159
age	0.0482589	0.0282474	1.708438	0.0877274
educ	-0.3453348	0.1947796	-1.772952	0.0764057
dem	15.4242556	1.0680327	14.441745	0.0000000
rep	-15.8495061	1.3113624	-12.086290	0.0000000

```r
#Calculating MSE
(mse_lm <- mean(linear$residuals^2))
```

```
## [1] 395.2702
```

Gender and party affiliation do have an impact on feeling towards Biden. A woman and a democrat will be more likely to have positive feelings towards Biden.

On the other hand, the results of age and education are not statistically significant at 0.05 significance level.

MSE is around 395.2701. We know that the closer MSE is towards zero, the better a model perfoms. Also, the R-squared is pretty low as well, meaning not much of the variation of the dependent variable is explained by the model. Therefore, we can say such a large number in scale may mean that the error might be large. However, since we are using the whole dataset for training, we have no idea of the predictive power of the model.

Question 2

```r
#Set seed
set.seed(222)

#Step 1: split the sample set
split <- initial_split(nes2008, prop = 0.5)
train <- training(split)
test <- testing(split)

#Step 2: Fit the model using only the training set
lm_training <- lm(biden~., data = train)
lm_training%>%
  tidy()%>%
  kable()
```

term	estimate	std.error	statistic	p.value
(Intercept)	58.1511462	4.4727490	13.0012094	0.0000000
female	6.3978310	1.3675224	4.6784105	0.0000033
age	-0.0225744	0.0404747	-0.5577414	0.5771600
educ	-0.1710101	0.2810940	-0.6083734	0.5430937
dem	16.6113166	1.5306762	10.8522736	0.0000000
rep	-15.2014117	1.8849430	-8.0646531	0.0000000

```r
#Step 3: Calculate the MSE using only the testing set
(mse_testing <- mean((test$biden - predict(lm_training, test))^2))
```

```
## [1] 385.9952
```

Step 4 Discuss the Result: Surprisingly, the MSE from the testing model is around 386, slightly smaller than that in question 1 which is 395. Usually MSE is expected to be greater as we half the observations in training the model. With fewer observations in training, the error tends to be greater. However, becasue of the randomness in setting seed and the fact that the process has only been repeated once, this unusual case could also occur.

Question 3

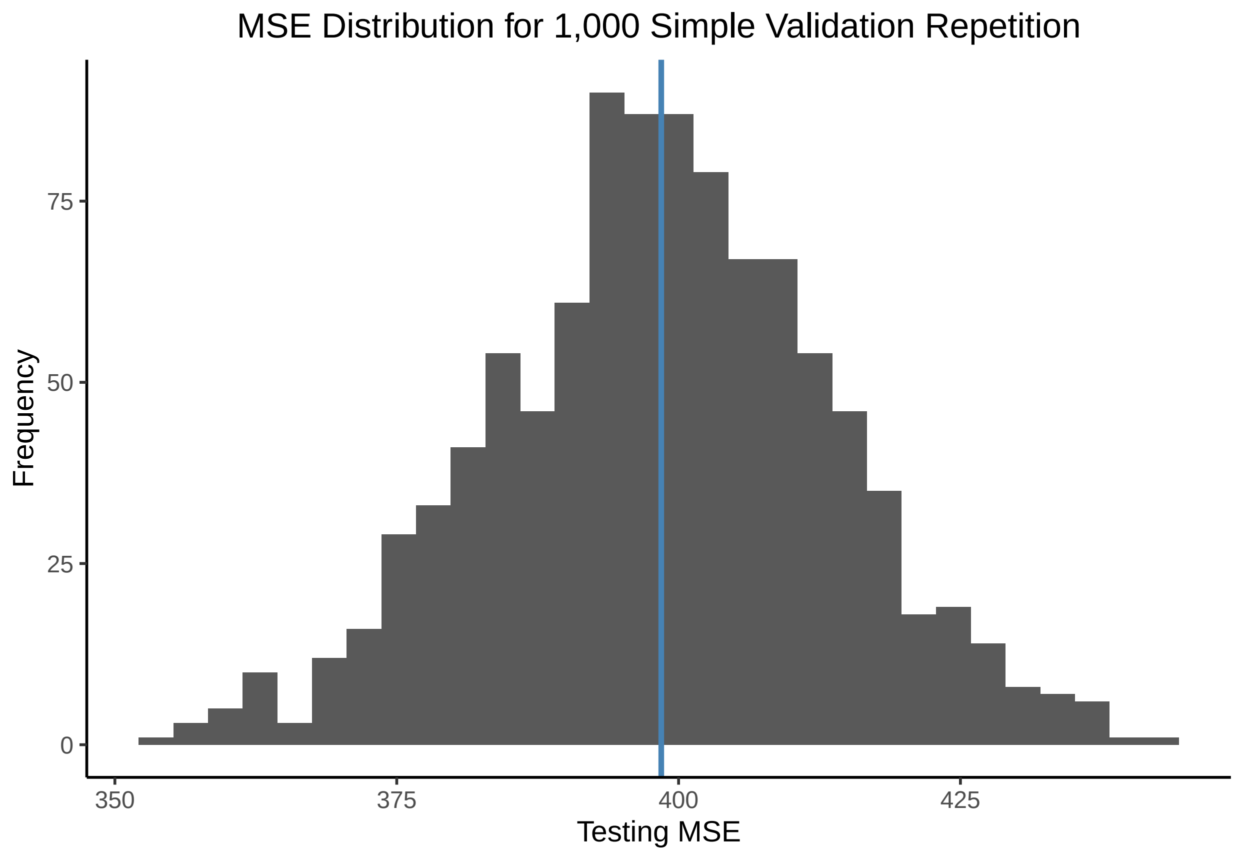
```r
#Set the number of repetition
#Set seed and number of repetitions
set.seed(333)
rep<- 1000
result<- c()

#Repeat for 1000 times
for (i in 1:rep){
#Split the sample set
  split <- initial_split(nes2008, prop = 0.5)
  train <- training(split)
  test <- testing(split)
  #Fit the model using only the training set
  lm_training <- lm(biden~., data = train)
  #Calculate the MSE using only the testing set
  mse_testing <- augment(lm_training, newdata = test)%>%
    mse(truth = biden, estimate = .fitted)
  result[i]<- mse_testing$.estimate[1]}

#Fit the result into a dataframe
result<- data.frame(mse = result)
(mean(result$mse))
```

```
## [1] 398.4621
```

```r
#Visualize the result
result %>%
  ggplot(aes(x = mse)) +
  geom_histogram()+
  geom_vline(aes(xintercept = mean(result$mse)),
             size = 1,
             color = "steelblue") +
  labs(x = "Testing MSE",
       y = "Frequency",
       title = "MSE Distribution for 1,000 Simple Validation Repetition")
```



The MSEs follow an approximately normal distribution with a mean of around 398.4621.

By comparing the values, we can see that this value is greater than the result we obtained from Question 1. This indeed is consistent with our speculation with Q2, a special case of random sampling. Normally, MSE tends to be higher with smaller sample size in the training dataset. We sometimes can obtain results like the one in Q2, but if we repeat the process for 1000 times and take the mean of MSE from the smaller sample size for the training set, we will probably obtain a relatively higher MSE, although the difference is not that huge in this case.

Question 4

```r
#Set seed
set.seed(444)

#Define the function
coefs <- function(splits, ...) {
  mod <- lm(..., data = analysis(splits))
  tidy(mod)
}

#Boostrap result
 boot_coef <- nes2008 %>%
   bootstraps(1000) %>%
   mutate(coef = map(splits, coefs,
                     as.formula(biden ~ .))) %>%
   unnest(coef) %>%
   group_by(term) %>%
   summarise(boot_estimate = mean(estimate),
             boot_se = sd(estimate, na.rm = TRUE))

#merge the two into one table
linear %>%
  tidy()%>%
  select(1:3)%>%
  rename(lm_estimate = estimate)%>%
  rename(lm_se = std.error)%>%
  left_join(boot_coef)%>%
  kable()
```

term	lm_estimate	lm_se	boot_estimate	boot_se
(Intercept)	58.8112590	3.1244366	58.8003048	3.0088653
female	4.1032301	0.9482286	4.1364762	0.9754381
age	0.0482589	0.0282474	0.0472202	0.0296139
educ	-0.3453348	0.1947796	-0.3415302	0.1874028
dem	15.4242556	1.0680327	15.3781038	1.0900558
rep	-15.8495061	1.3113624	-15.8631996	1.3709421

There is not much difference in terms of numeric outputs between the linear regression and the bootstrap, though bootstrap generally has larger standard errors and smaller coefficients across predictors. Such a small difference show that our estimate in Question 1 might be accurate.

As for these two methods, both are estimating the true population by fitting the samples into linear regression lines. But since the training set is halved in bootstrap case, the error of the regression line for the bootstrap might be greater.

As for the bootstrap method in general, it is simply resampling from the limited sample available and repeat the process (with replacement and the same sample size) for numerous times. And we will examine the distribution of the estimator we focus to estimate the population.

By using bootstrap, we don't have to care about the homoskedasticity of error terms nor any distributional assumption of the error terms, which linear regression method would consider to ensure the standard errors to be accurately estimated. Therefore, we can say that bootstrap method is much more robust in estimating how close our sample is to the true population when we have no idea of the true population shape.

In this regard, the sample size is halved with the bootstrap approach and as we are resampling with 1000 times its distribution is large enough to be much closer to the population distribution.