BattishaPset2

```
#Import Libaries
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------

## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ------------------------------------------------------------------------------------ ti
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(rsample)
library(broom)
library(boot)
library(rcfss)
library(yardstick)
```

```
## For binary classification, the first factor level is assumed to be the event.
## Set the global option `yardstick.event_first` to `FALSE` to change this.

##
## Attaching package: 'yardstick'

## The following object is masked from 'package:readr':
##
##     spec
```

```
#Read File
biden <- read_csv("nes2008.csv")
```

```
## Parsed with column specification:
## cols(
##   biden = col_double(),
##   female = col_double(),
##   age = col_double(),
##   educ = col_double(),
##   dem = col_double(),
##   rep = col_double()
## )
```

```
biden
```

```
## # A tibble: 1,807 x 6
##    biden female   age  educ   dem   rep
##    <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
##  1    90      0    19    12     1     0
##  2    70      1    51    14     1     0
##  3    60      0    27    14     0     0
##  4    50      1    43    14     1     0
##  5    60      1    38    14     0     1
##  6    85      1    27    16     1     0
##  7    60      1    28    12     0     0
##  8    50      0    31    15     1     0
##  9    50      1    32    13     0     0
## 10    70      0    51    14     1     0
## # ... with 1,797 more rows
```

#Question 1

```
#Question 1: MSE of entire dataset

#Fit model to entire dataset
lm_biden <- lm(biden ~ female+age+educ+dem+rep,data=biden)
summary(lm_biden)
```

```
##
## Call:
## lm(formula = biden ~ female + age + educ + dem + rep, data = biden)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -75.546 -11.295   1.018  12.776  53.977
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  58.81126    3.12444  18.823  < 2e-16 ***
## female        4.10323    0.94823   4.327 1.59e-05 ***
## age           0.04826    0.02825   1.708   0.0877 .
## educ         -0.34533    0.19478  -1.773   0.0764 .
## dem          15.42426    1.06803  14.442  < 2e-16 ***
## rep         -15.84951    1.31136 -12.086  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.91 on 1801 degrees of freedom
## Multiple R-squared:  0.2815, Adjusted R-squared:  0.2795
## F-statistic: 141.1 on 5 and 1801 DF,  p-value: < 2.2e-16
```

```
#Predict values of entire dataset using model
biden_prediction <- augment(lm_biden, newdata = biden)

#Calculate MSE
```

```
mse_biden <- biden_prediction %>%
  mse(truth = biden, estimate =.fitted)
(mse_biden_value <-mse_biden$.estimate)
```

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

Question 1: Discuss Results The Mean Squared Error between the actual and the predicted values was 395.2702. The multiple R-squared value was 0.2815. The low R^2 value and the high MSE (especially given that Biden scores only go from 0 to 100) show that our model isn't doing the best job predicting the Biden scores.

The linear regression model on the entire dataset estimated coefficients of absolute value less than 1 for age and education, around 4 for female, and of absolute value aruond 15 for democrat and republican alignment. This shows that age and education were minimal predictors, while democratic and republicatn alignment were major predictors and sex was a medium predictor. Interestingly, it estimated that republican alignment had a slightly greater coefficient than democrat alignment.

The output also gave significantly smaller T and P Values for age and education than for the other factors, showing that age and education coefficients were not as statistically significant as the rest, and are not likely to be as useful predictors.

#Question 2

```
#Question 2.1: Split the sample set into a training set (50%) and a holdout set (50%)

#sample dataset
set.seed(24)

#sampling <- sample(nrow(biden), nrow(biden)*.5, replace=FALSE)
sampling <-  initial_split(data=biden, prop=0.5)

#create training dataset from second of randomized dataset
training <- training(sampling)

#create holdout dataset from first half of randomized dataset
testing <- testing(sampling)

#confirm that dataset sizes are correct
nrow(testing)
```

```
## [1] 903
```

```
nrow(training)
```

```
## [1] 904
```

```
nrow(biden)
```

```
## [1] 1807
```

```r
#Question 2.2: Fit the linear regression model using only the training observations

lm_training <- lm(biden ~ female+age+educ+dem+rep,data=training)
summary(lm_training)
```

```
##
## Call:
## lm(formula = biden ~ female + age + educ + dem + rep, data = training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -74.598 -11.135   1.386  12.014  44.488
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  51.29530    4.31962  11.875   <2e-16 ***
## female        3.39729    1.35202   2.513   0.0122 *
## age           0.06818    0.03981   1.713   0.0871 .
## educ          0.24344    0.27014   0.901   0.3677
## dem          15.07483    1.51920   9.923   <2e-16 ***
## rep         -17.02157    1.87146  -9.095   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.93 on 898 degrees of freedom
## Multiple R-squared:  0.2813, Adjusted R-squared:  0.2773
## F-statistic: 70.31 on 5 and 898 DF,  p-value: < 2.2e-16
```

```r
#Question 2.3: Calculate the MSE using only the test set observations.

#Predict values of testing dataset using model based on training dataset
testing_prediction<- augment(lm_training, newdata = testing)

#Calculate MSE
mse_test <- testing_prediction %>%
    mse(truth=testing$biden, estimate=.fitted)
(mse_test_value <- mse_test$.estimate)
```

```
## [1] 400.9878
```

```r
#Question 2.4: Compare MSE values between Question 2 and Question 1

(mse_difference <- abs(mse_test_value-mse_biden_value))
```

```
## [1] 5.717639
```

```r
(mse_ratio <-mse_test_value/mse_biden_value)
```

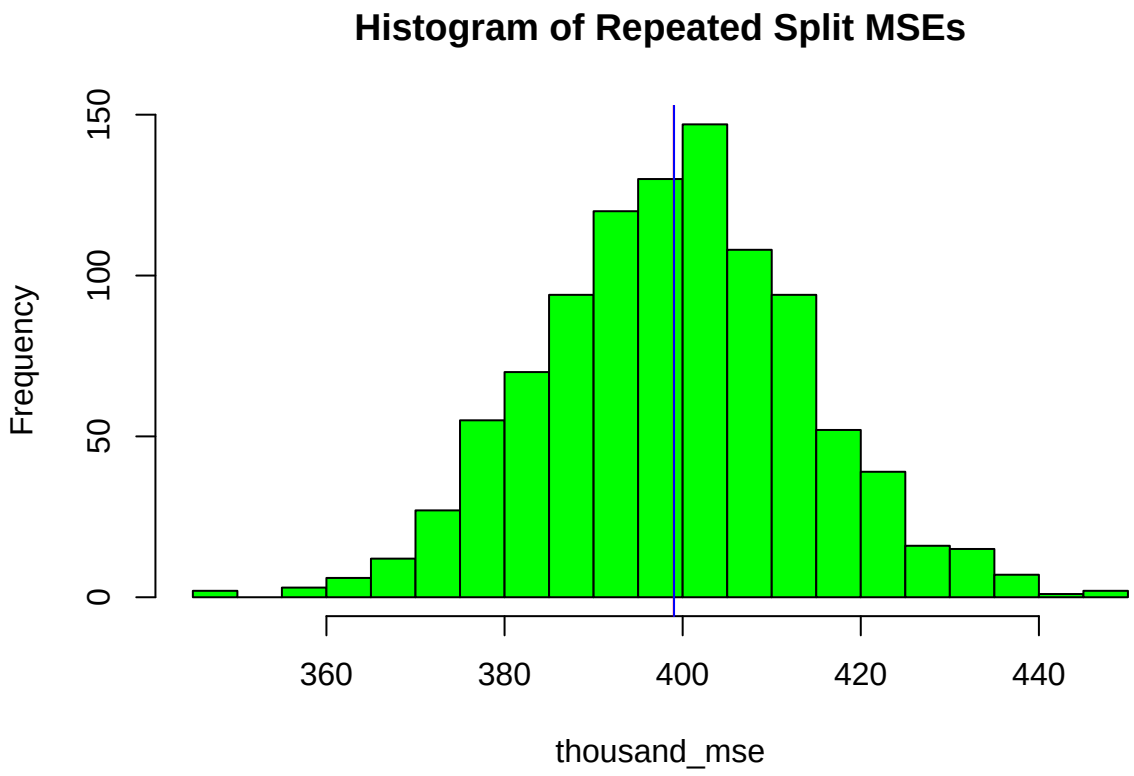
```
## [1] 1.014465
```

Question 2: Compare to MSE from Q1 While in Question 1 the MSE based on the entire dataset was 395.27, in Question 2 the MSE based on the test set was 400.9878. This is a relatively small change in MSE, with the MSE increasing by 5.717, or 1.447% in Question 2. The small 1.447% change in MSE tells us that we are probably not overfitting our model to our data.

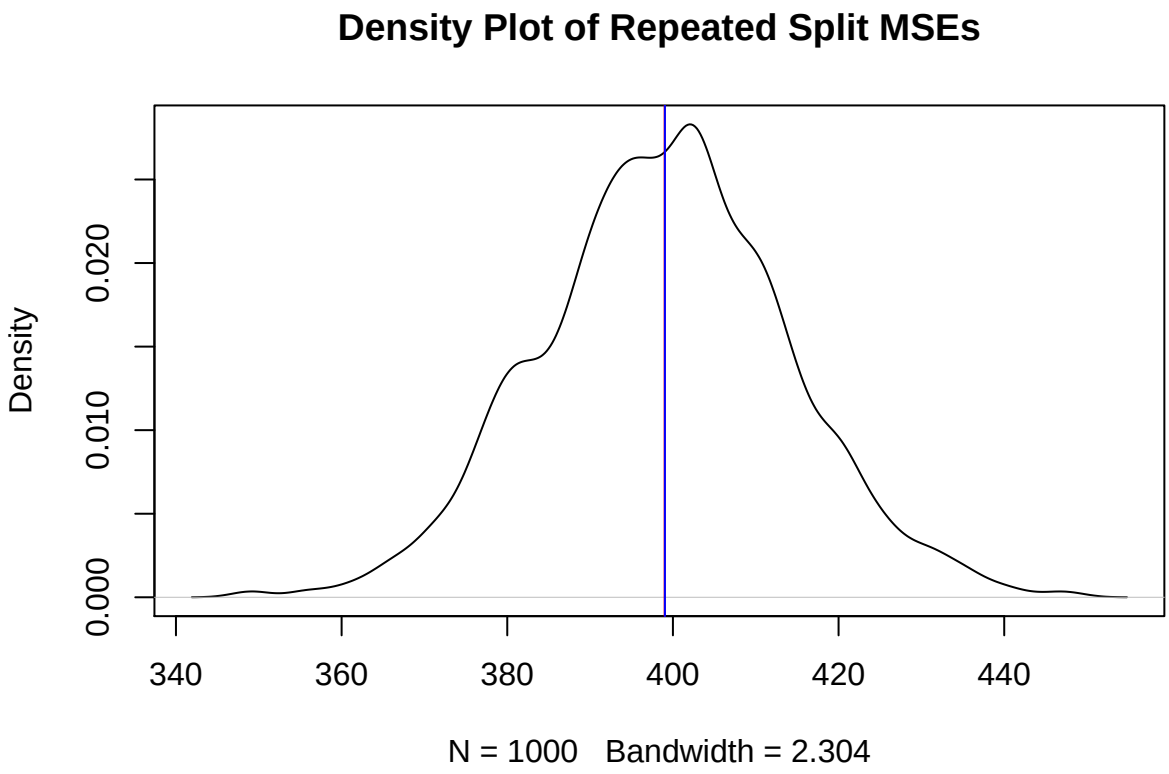
#Question 3

```
#Question 3: Repeating Simple Validation 1000 times
thousand_mse <- c()
for (i in c(1:1000)){
  #sampling <- sample(nrow(biden), nrow(biden)*.5, replace=FALSE)
  loop_sample<-  initial_split(data=biden, prop=0.5)

  #create training dataset from second of randomized dataset
  loop_train <- training(loop_sample)

  #create holdout dataset from first half of randomized dataset
  loop_test <- testing(loop_sample)

  #Fit the linear regression model using only the training observations
  loop_lm <- glm(biden ~ female+age+educ+dem+rep,data=loop_train)

  #Predict values of testing dataset using model based on training dataset
  loop_prediction<- augment(loop_lm, newdata = loop_test)

  #Calculate MSE
  loop_mse_test <- loop_prediction %>%
      mse(truth=loop_test$biden, estimate=.fitted)
  loop_mse_value <- loop_mse_test$.estimate

  #Append to loop
  thousand_mse <- append(thousand_mse, loop_mse_value)
}
```

```
#Question 3.1: Graph Repeated Values

#Graph Histogram with Mean in Red
hist(thousand_mse, main="Histogram of Repeated Split MSEs", breaks=20, col="green")
abline(v=mean(thousand_mse),col="red")
abline(v=median(thousand_mse),col="blue")
```




```
#Graph Density with Mean in Red
plot(density(thousand_mse, adjust=.7), main="Density Plot of Repeated Split MSEs", col = "black")
abline(v=mean(thousand_mse),col="red")
abline(v=median(thousand_mse),col="blue")
```




```
#Output Summary of Data
summary(thousand_mse)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   348.8   389.4   399.0   399.0   409.0   447.9
```

Question 3: Comment on Graphs The Density plot shows that the distribution is shaped relatively like a bell-curve, with a slight downturn around 380 and a spike around 405. This means that there is something in the numbers of the data that is generating less MSE around 380 than expeccted and more MSEs around 405 than expected.

The Histogram confirms the notion that there are more MSEs from 400-405 than expected; while normally, the frequency would start decreasing after hitting the mean value, here it increases in the 400-405 bar, before decreasing more sharply than expected in the 405-410 bar. This shows that there is something in the data that is favoring an MSE of 400-405 over an MSE of 405-410.

The Summary of the data shows that the MSEs range from 348.8 to 447.9 (roughly +-50 from the Median), and are centered around a mean & median of 399. The IQR of the data is from 389.4 to 409 (so roughly, +-10 from the Median). Overall, the range of the MSEs is around 100, and the IQR is around 20.

```
#Question 4: Bootstrapping Samples


# bootstrapped estimates of the parameter estimates and standard errors
lm_boot <- function(splits, ...) {
  ## use `analysis` or `as.data.frame` to get the analysis data
```

```
  mod <- lm(..., data = analysis(splits))
  tidy(mod)
}

biden_boot <- biden %>%
  bootstraps(1000) %>%
  mutate(coef = map(splits, lm_boot, as.formula(biden ~ female+age+educ+dem+rep)))

biden_boot %>%
  unnest(coef) %>%
  group_by(term) %>%
  summarize(.estimate = mean(estimate),
            .se = sd(estimate, na.rm = TRUE))
```

```
## # A tibble: 6 x 3
##   term        .estimate    .se
##   <chr>           <dbl>  <dbl>
## 1 (Intercept)   58.6    3.03
## 2 age            0.0491 0.0290
## 3 dem           15.4    1.07
## 4 educ          -0.335  0.193
## 5 female         4.09   0.951
## 6 rep          -15.9    1.37
```

```
#Populate Question 1 Results for Comparision Purposes:
summary(lm_biden)
```

```
##
## Call:
## lm(formula = biden ~ female + age + educ + dem + rep, data = biden)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -75.546 -11.295   1.018  12.776  53.977
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  58.81126    3.12444  18.823  < 2e-16 ***
## female        4.10323    0.94823   4.327 1.59e-05 ***
## age           0.04826    0.02825   1.708   0.0877 .
## educ         -0.34533    0.19478  -1.773   0.0764 .
## dem          15.42426    1.06803  14.442  < 2e-16 ***
## rep         -15.84951    1.31136 -12.086  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.91 on 1801 degrees of freedom
## Multiple R-squared:  0.2815, Adjusted R-squared:  0.2795
## F-statistic: 141.1 on 5 and 1801 DF,  p-value: < 2.2e-16
```

Question 4: Compare Bootstrapped output with Q1 Output The mean bootstrapped output and the Q1 output present relatively similar estimates and standard errors, which is unsuprising given that the distribution of

the bootstrap generally revolves around the mean of the dataset being bootstrapped from. There are however some slight, but interesting differences. For one, the coefficient for republican alignment is greater in the bootstrap, while the coefficient for democratic alignment is smaller. Addtionally, the standard error for the age parameter is greater in the bootstrap than in the original output.

Conceptually, what we've done here is sampled rows with replacement from the biden dataset to create 1000 similar datasets. We've then ran a linear regression on each of these datasets, and averaged their coefficients and standard errors to find the .estimate and .se values. In terms of usage, bootstrapping is typically utilized to determine if our assumptions about a model distribution are correct, or to find confidence intervals for a dataset that would include the population values even if our assumptions were incorrect. In general, bootstrapping helps us increase our number of samples so we can generalize about a population when we have a small sample size.

In this particular case, we've used bootstrapping to see if the coefficients and errors of our linear model would vary greatly based on the samples chosen. By findnig the mean values of the 1000 samples, we have determined that the average of our resampling does not vary greatly from our original values.