John_Kim_PS2

John Kim

2020 02 02

1. Estimate the MSE of the model using the traditional approach. That is, fit the linear regression model using the entire dataset and calculate the mean squared error for the entire dataset. Present and discuss your results at a simple, high level.

The mean squared error (MSE) of the model using linear regression with the entire data set is 395.2702. This means that the mean absolute value of the error is 19.88, and this can be considered a low number regarding that it is smaller than the standard deviation of the biden variable (23.46).

```
rm(list=ls())
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## Warning: package 'purrr' was built under R version 3.6.2
```

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

```
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.6.2
```

```
library(broom)
library(rsample)
```

```
## Warning: package 'rsample' was built under R version 3.6.2
```

```
library(rcfss)
library(yardstick)
```

```
## Warning: package 'yardstick' was built under R version 3.6.2
```

```
## 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
```

```
library(Metrics)
```

```
## Warning: package 'Metrics' was built under R version 3.6.2
```

```
##
## Attaching package: 'Metrics'
```

```
## The following objects are masked from 'package:yardstick':
##
##     accuracy, mae, mape, mase, precision, recall, rmse, smape
```

```
## The following object is masked from 'package:rcfss':
##
##     mse
```

```
df <- read.csv(file.choose())
df <- as_tibble(df)

bidenglm <- glm(biden ~ female + age + educ + dem + rep, data = df)
mse(bidenglm$fitted.values, df$biden)
```

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

```
summary(bidenglm)
```

```
##
## Call:
## glm(formula = biden ~ female + age + educ + dem + rep, data = df)
##
## Deviance 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
##
## (Dispersion parameter for gaussian family taken to be 396.587)
##
##     Null deviance: 994144  on 1806  degrees of freedom
## Residual deviance: 714253  on 1801  degrees of freedom
## AIC: 15947
##
## Number of Fisher Scoring iterations: 2
```

```
# 395.2702
sd(df$biden)
```

```
## [1] 23.46203
```

2. Calculate the test MSE of the model using the simple holdout validation approach.

The MSE of the population regression is 395.2702, and the MSE using the test set is 400.8075. There is no big difference between the two which signifies that the model in the question above does not show the

overfitting problem.

```
# Split the sample set into a training set (50%) and a holdout set (50%). Be sure to set
# your seed prior to this part of your code to guarantee reproducibility of results.
set.seed(1111)
df_split <- initial_split(data = df,
                          prop = 0.5)
df_train <- training(df_split)
df_test <- testing(df_split)
df_lm <- glm(biden ~ female + age + educ + dem + rep, data = df)
summary(df_lm)
```

```
##
## Call:
## glm(formula = biden ~ female + age + educ + dem + rep, data = df)
##
## Deviance 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
##
## (Dispersion parameter for gaussian family taken to be 396.587)
##
##     Null deviance: 994144  on 1806  degrees of freedom
## Residual deviance: 714253  on 1801  degrees of freedom
## AIC: 15947
##
## Number of Fisher Scoring iterations: 2
```

```
mse(df_lm$fitted.values, df$biden)
```

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

```
# Fit the linear regression model using only the training observations.

(train_mse <- augment(df_lm, newdata = df_train) %>%
  rcfss::mse(truth = biden, estimate = .fitted))
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 mse     standard        390.
```

```
# Calculate the MSE using only the test set observations.

(test_mse <- augment(df_lm, newdata = df_test) %>%
    rcfss::mse(truth = biden, estimate = .fitted)) #400.8075
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 mse     standard        401.
```

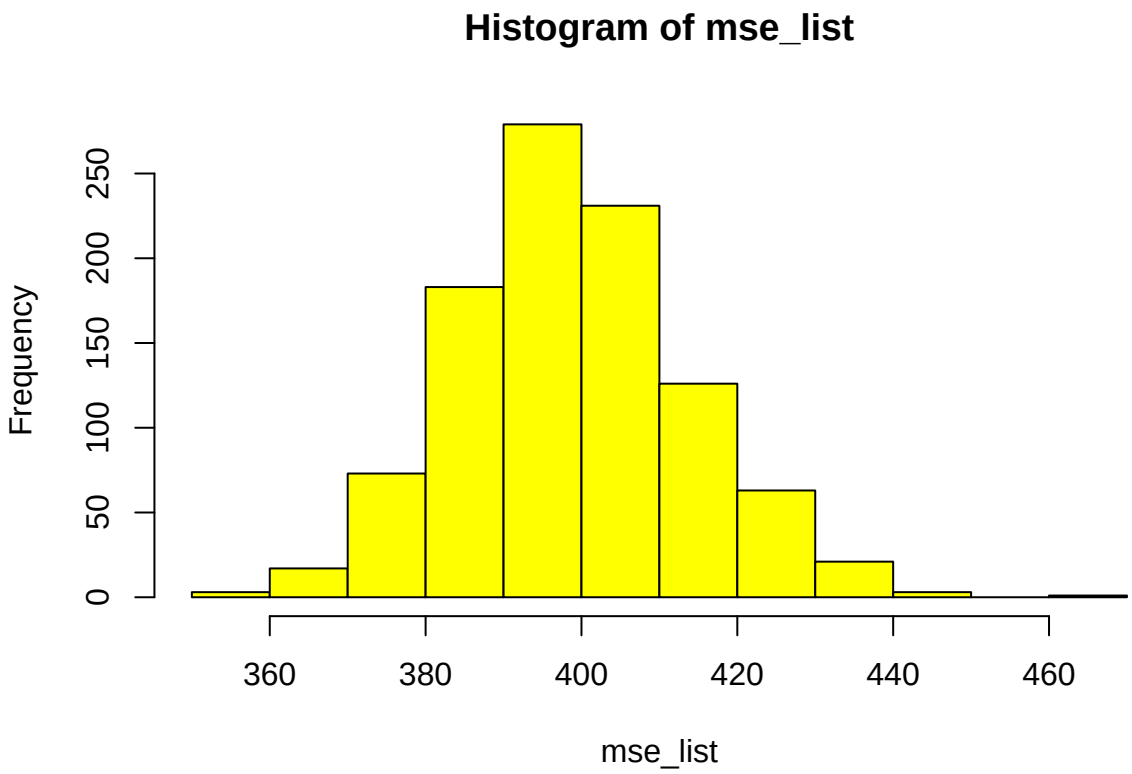
```
# How does this value compare to the training MSE from question 1? Present numeric
# comparison and discuss a bit.
```

3. Repeat the simple validation set approach from the previous question 1000 times, using 1000 different splits of the observations into a training set and a test/validation set. Visualize your results as a sampling distribution (hint: think histogram or density plots). Comment on the results obtained.

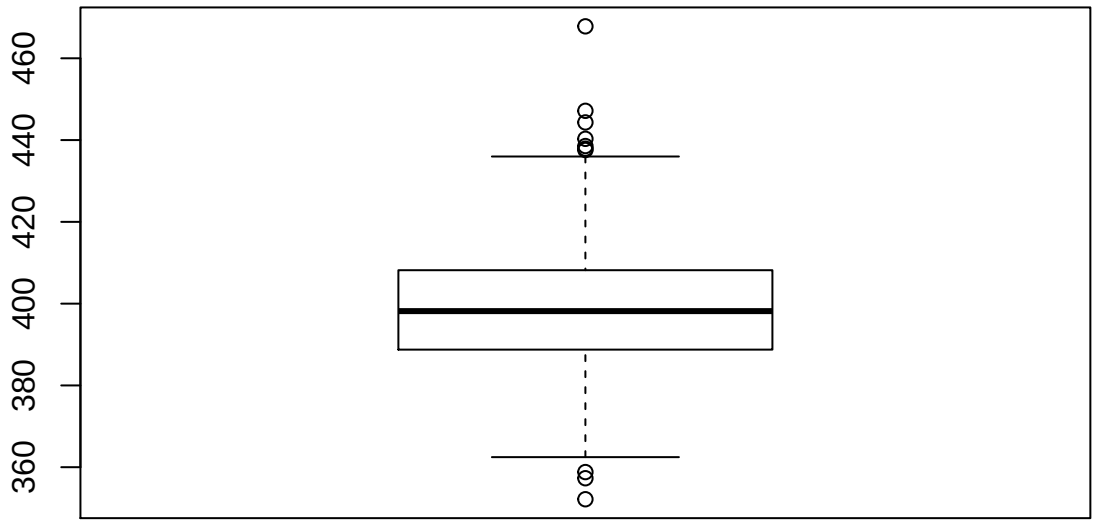
The mean of the sampling distribution from 1000 samples is 398.9871. This is still larger than that of the originial model, but it is less than the MSE attained from the model using the test data set. This implies that calculating the mean MSE from drawing a lot of samples may bring down the MSE level similar to the population level.

```
set.seed(89)
nrep = 1000
prop = 0.5
mse_list = c() # initialize empty vector to save MSE for each iteration
for (i in 1:nrep){
  df_train = sample(1:nrow(df), prop*nrow(df))
  df_test = setdiff(1:nrow(df), df_train)
  df_lm = glm(biden ~ female + age + educ + dem + rep, data = df[df_train, ])
  pre_biden = predict(df_lm, newx = df[df_test, ])
  mse_list[i] = mean((df$biden-predict(df_lm, df))[df_test]^2)
}

hist(mse_list, col = "yellow")
```




```
boxplot(mse_list)
```




```
mean(mse_list)
```

```
## [1] 398.9871
```

4. Compare the estimated parameters and standard errors from the original model in question 1 (the model estimated using all of the available data) to parameters and standard errors estimated using the bootstrap (B = 1000). Comparison should include, at a minimum, both numeric output as well as discussion on differences, similarities, etc. Talk also about the conceptual use and impact of bootstrapping.

Bootstrapping is drawing repeated samples with replacement. That enables us to estimate the uncertain parameters accurately. The intercept of the original model was 58.81126, and that of the bootstrap model is 58.81125899. The coefficient comparison for all the variables in the population regression and the bootstrapping models are female: 4.10323, 4.10323009; age: 0.04826, 0.04825892; educ: -0.34533, -0.34533479; dem: 15.42426, 15.42425563; rep: -15.84951, -15.84950614.

```
library(boot)
```

```
## Warning: package 'boot' was built under R version 3.6.2
```

```
set.seed(88)
boot_biden <- function(data, df_test) +
  return(coef(lm(biden ~ female + age + educ + dem + rep, data = df, subset = df_test)))
(bootbd <- boot(df, boot_biden, 1000))
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
```

```
## Call:
## boot(data = df, statistic = boot_biden, R = 1000)
##
##
## Bootstrap Statistics :
##         original        bias    std. error
## t1*  58.81125899 -0.0739438614  3.08321875
## t2*   4.10323009 -0.0137159539  0.98012865
## t3*   0.04825892  0.0014492689  0.02896875
## t4*  -0.34533479  0.0003907533  0.19480597
## t5*  15.42425563  0.0271400601  1.04183640
## t6* -15.84950614  0.0442995022  1.34057279
```