Application and Analysis of Bootstrapping Methods on a Walmart Dataset

Sirpreet "Shawn" Padda
California State University, East Bay
STAT 641
Dr. Jiyoun Myung
March 11, 2022

Table of Contents

Introduction

Walmart is one of the most well-known retail chains in the United States. Weekly sales vary depending on holidays and special events. This report uses a dataset from Walmart that pertains to Walmart's weekly sales; regression analysis and applied bootstrapping methods were applied on the dataset. There are two key research questions. The first research question is as follows: "What is the relationship between the weekly sales as the response and all other chosen explanatory predictors?" This report will further expand on the first question by comparing the standard errors, and 95% confidence interval of the classical fitted regression model with the residual bootstrap resampling and observation bootstrap resampling regression methods. The second research question pertains to conducting hypothesis testing using the response (weekly sales) and the factor (holiday flag) with an infer package.

The motivation for this project is based on the fact that linearity, independence, normality, and equal variance (LINE) assumptions are frequently violated. People enjoy transforming data or removing extreme outliers. Bootstrapping has the virtue of not relying on assumptions; it is the most appropriate way to combat violations and outliers in the dataset.

The goal of data analysis on this dataset is to apply different bootstrapping methods on a multiple linear regression model in R. In addition to this, providing an adequate amount of information to the reader along with appropriate interpretations was also taken into account. An additional goal of data analysis was to check the LINE assumptions. If there were any violations of LINE assumptions, then bootstrapping methods were considered.

Data Description

Walmart's weekly sales are often impacted by unseen demand during special events and holidays, with the four biggest events and holidays being the Super Bowl, Labor Day, Thanksgiving, and Christmas. Walmart runs several promotional markdown events during the year. These markdowns coincide with major holidays and events. The weighting of weeks that included these holidays was five times higher than non-holiday weeks. The dataset covers sales data of 45 Walmart stores from different regions in the United States with historical sales data between February 5th, 2010, and November 11th, 2012. There are no "NA" values in the dataset. There are extreme outliers in the dataset, which could affect the "LINE" assumptions. The dimensions of the dataset include 6435*8. The dataset is obtained from Kaggle.

Variable Description

The dataset contains a total of 8 variables. They are as follows: store (the store number), date (the week of sales), weekly sales (sales for the given store), holiday flag (whether the week is a special holiday week, holiday week - 1 and non-holiday week - 0), temperature (temperature on the day of the sale), Fuel_Price (cost of fuel in the region), CPI (prevailing consumer price index), and unemployment (prevailing unemployment rate). This report only considers the relevant variables to fit the multiple linear regression model for regression purposes. Weekly sales are used as the response variable and other variables such as temperature, fuel price, CPI, and unemployment are used as the predictor variables. The holiday flag was used against the response variable to address the second question. The remaining variables were unused in this project.

Exploratory Data Analysis

In Table 1, the summary() function was used to print the summary statistic of the fitted model. I discovered that the p-value is very small, and all the predictors are significant. Moreover, I found about 2.3% of the variability in y (weekly sales) is explained by the multiple linear regression model. As a result of the lower adjusted R-squared, the multiple linear regression model does not appear to be a good fit.

Table1: Summary Table

```
Call:
lm(formula = Weekly_Sales ~ Temperature + CPI + Unemployment,
    data = Walmart.2)

Residuals:
    Min      1Q  Median      3Q     Max
-953444 -478699 -114424  396547 2789840

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1697086.8    52433.2  32.367  < 2e-16 ***
Temperature     -947.1      388.2  -2.439   0.0147 *
CPI            -1550.8      189.9  -8.167 3.77e-16 ***
Unemployment  -40827.9     3941.2 -10.359  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 557600 on 6431 degrees of freedom
Multiple R-squared:  0.02424,   Adjusted R-squared:  0.02378
F-statistic: 53.25 on 3 and 6431 DF,  p-value: < 2.2e-16
```

Data Visualization

In Figure 1, I used the correlation matrix plot to indicate the perfect negative and positive linear correlation between the response (weekly sales) and other predictor variables like temperature, CPI, and unemployment. In Figure 2, I used the logarithmic scale to better visualize the relationship between weekly sales and temperature. The scatter plot shows a negative relationship between weekly sales and temperature using a logarithmic scale. As the temperature rises, the weekly sales tend to decline. The colors of the points are associated with the levels of a holiday flag variable in the legend.

Figure 1: Correlation Matrix

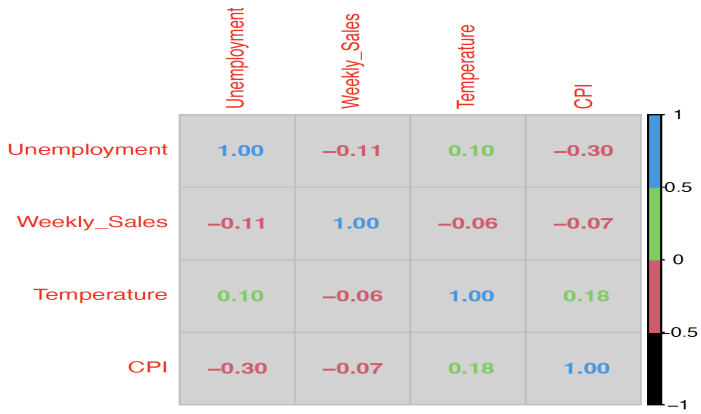


Figure 2: Scatter plot

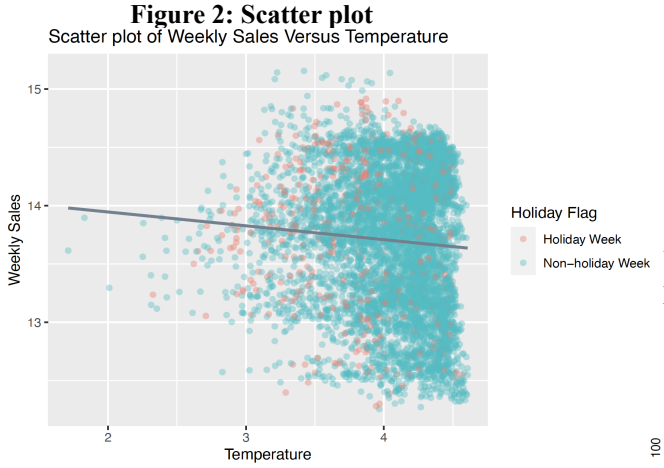
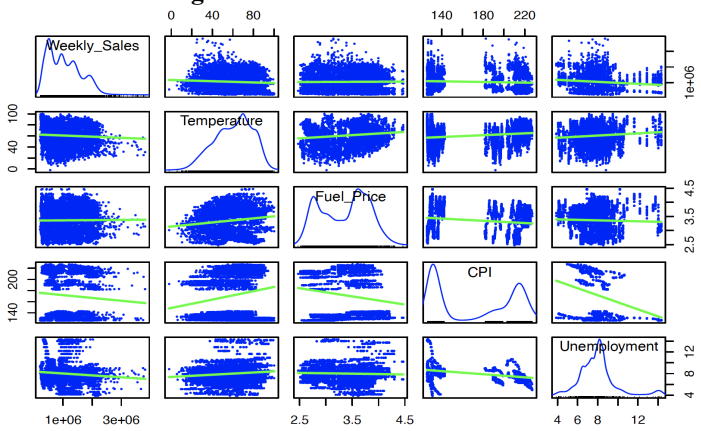


Table 2: 95% Confidence Interval

	2.5 %	97.5 %
(Intercept)	1594300.328	1799873.2817
Temperature	-1708.140	-185.9982
CPI	-1923.016	-1178.5425
Unemployment	-48554.086	-33101.8019

In Figure 3, I utilized the scatter plot matrix to understand the marginal relationship among relevant variables in the dataset. The scatter plot matrix depicts that few variables have a strong positive or negative relationship. However, the remaining variables showed a neutral, slight negative, and slight positive linear relationship with the response. The smooth density curve indicates that the data points are extremely varied.

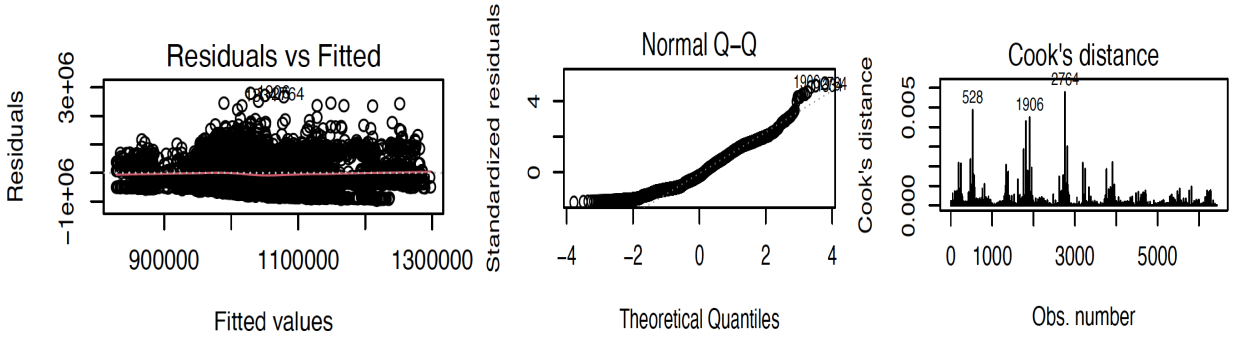
Figure 3: Scatter Plot Matrix



Methods and Results

I commenced the project with regression analysis. The multiple linear regression model took into account all variables. Then, to get my final model, I utilized a backward stepwise selection technique with AIC. I removed the Fuel_Price variable. I refitted the linear model. I ran a diagnostic check to check LINE assumptions. As we can see in Figure 4, the linearity and constant variance assumptions were adequately satisfied, but I found violations in the normality assumption. Some extreme outliers were detected in the data. The Cook's distance indicates a 2764th data point, which is influential.

Figure 4: Diagnostic Plots



Subsequently, I considered applying two common bootstrap regression techniques: (1) Random x resampling or observation resampling (2) Fixed x resampling or residual resampling.

Observation Resampling Technique

First, I used the observation resampling technique. The bootstrap statistic in summary Table 2 reveals the bootstrap estimates, their bias, and their standard errors. Some of my major findings are discussed below.

Findings

Comparing the bootstrap estimates to the estimates of the traditional linear model (see Table 1), the bootstrap estimates are identical. However, upon comparing the standard errors of the bootstrap estimates to the standard errors of the fitted multiple linear regression model, I discovered in Table 3, a couple of the bootstrap estimates for the coefficients have smaller standard errors while others have a slightly bigger standard error than the regular/traditional linear model.

Table 3: Bootstrap Statistics Summary Table

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Walmart.2, statistic = boot_c, R = 8000)


Bootstrap Statistics :
        original      bias    std. error
t1* 1697086.8051 376.227923  52230.7502
t2*    -947.0689   2.299452    403.4339
t3*   -1550.7795  -1.564291    202.9305
t4*  -40827.9441 -20.667163   3603.7490
```

Visualization

I visualized the bootstrap replicates. The results are shown on the right (in Figure 5 and 6) for indexing positions 2 and 3 (Indexing position 2 represents βˆ1, Indexing position 3 represents βˆ2) appears pretty normal. The histogram plot shows the symmetric distribution of the bootstrap replicates for βˆ1 and βˆ2.

Figure 5: Bootstrap Replicate Plot (When Index = 2)

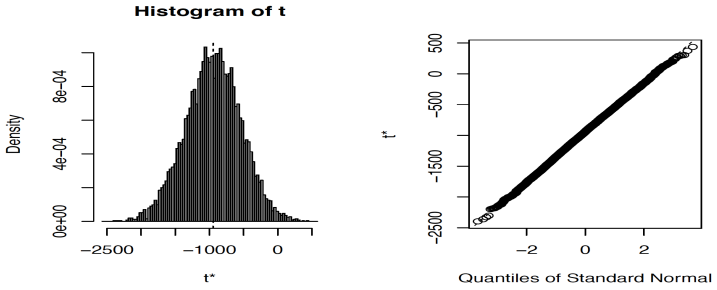
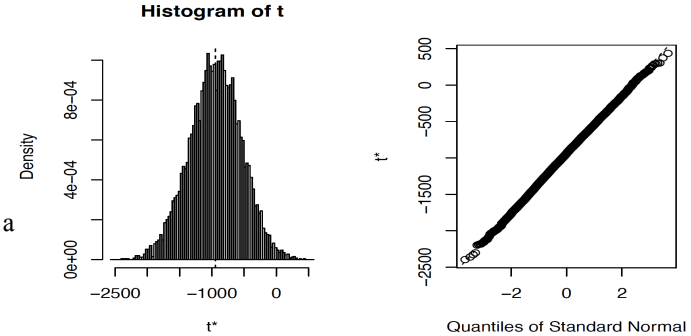


Figure 6: Bootstrap Replicate Plot (When Index = 3)



95% Bootstrap Confidence Interval

I contrasted the 95% bootstrap confidence interval results of indexing position 3 (indicating CPI) with a traditional 95% confidence interval of the linear model (see Table 2). In Table 4, I noticed that the 95% normal, percentile, and BCa confidence intervals have a narrower range than the regular model 95% confidence interval. However, the BCa 95% confidence

Table 4: Bootstrap 95% Confidence Interval

```
CALL :
boot.ci(boot.out = boot.random, conf = 0.95, type = c("norm",
    "perc", "bca"), index = 3)

Intervals :
Level      Normal             Percentile            BCa
95%   (-1947, -1151 )   (-1947, -1152 )   (-1941, -1148 )
Calculations and Intervals on Original Scale
```

interval has a slightly wider range than the other two confidence intervals. The values of all the bootstrap confidence intervals are subtly different from the regular model. Although the normal and percentile confidence intervals have a narrower range and provide more precise bootstrap estimates, I would pick a BCa bias-corrected 95% confidence interval which mitigates the risk of skewness and bias. For instance, in order to interpret 95% BCa C.I, we are 95% confident that the true value of the bootstrap replicates for the coefficient β^2 is between -1941 and -1148.

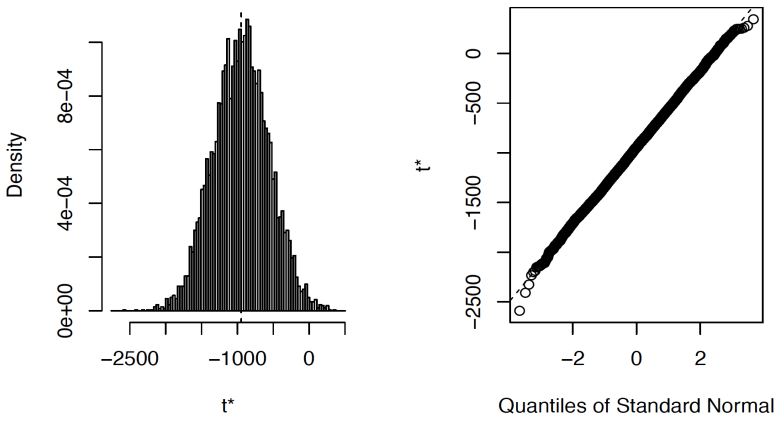
Table 5: Bootstrap Statistic Summary Table

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Walmart.2, statistic = boot_fun_fixed, R = 8000)


Bootstrap Statistics :
        original       bias    std. error
t1* 1697086.8051 -46.07332060  51820.7758
t2*    -947.0689  -1.42025917    388.6891
t3*   -1550.7795  -0.03265342    189.8264
t4*  -40827.9441  26.40920305   3891.6983
```

Figure 7: Bootstrap Replicate Plot (When Index = 2)



Residual Resampling Technique

Second, I utilized the residual resampling technique. In summary Table 5, the bootstrap statistic displays the bootstrap estimates, their bias, and their standard errors. Some of my major findings are discussed below.

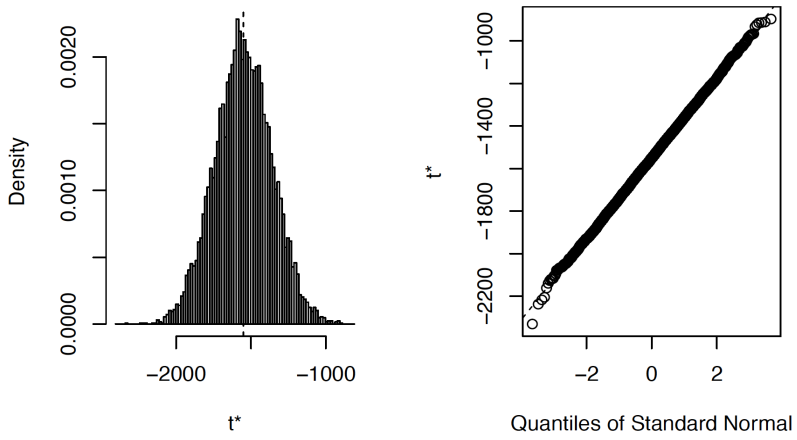
Findings

While comparing bootstrap estimates to traditional linear model estimates (see Table 1), the bootstrap estimates are identical. However, when the standard errors of the bootstrap estimates are compared to the standard errors of the fitted multiple linear regression model, I noticed that the bootstrap estimations have lower standard error than the traditional linear model.

Visualizations

The bootstrap replicates were presented graphically in Figures 7 and 8. The results are shown in the right for indexing positions 2 and 3 (Indexing position 2 represents β^1, Indexing position 3 represents β^2) appear to be fairly normal. The histogram plot shows the symmetric distribution of the bootstrap replicates for β^1 and β^2.

Figure 8: Bootstrap Replicate Plot (When Index = 3)



95% Bootstrap Confidence Interval

I compared the 95% bootstrap confidence interval results of indexing position 3 (indicating CPI) with a traditional 95% confidence interval of the linear model (see Table 2). In Table 6, I noticed that

Table 6: Bootstrap 95% confidence interval

```
CALL :
boot.ci(boot.out = boot.fixed, conf = 0.95, type = c("norm",
    "perc", "bca"), index = 3)

Intervals :
Level      Normal             Percentile            BCa
95%   (-1923, -1179 )   (-1925, -1185 )   (-1924, -1185 )
Calculations and Intervals on Original Scale
```

the 95% normal bootstrap confidence values are nearly identical to the traditional 95% confidence interval. Moreover, the 95% percentile bootstrap confidence interval has a wider range than the traditional 95% confidence interval. However, the values are subtly different. Furthermore, the 95% bias-corrected and accelerated (BCa) bootstrap confidence interval values vary substantially from the regular 95% confidence interval. The BCa 95% confidence interval produced a narrower range than the 95% percentile bootstrap confidence interval. For instance, in order to interpret 95% BCa C.I, we are 95% confident that the true value of the bootstrap replicates for the coefficient βˆ2 is between -1924 and -1185.

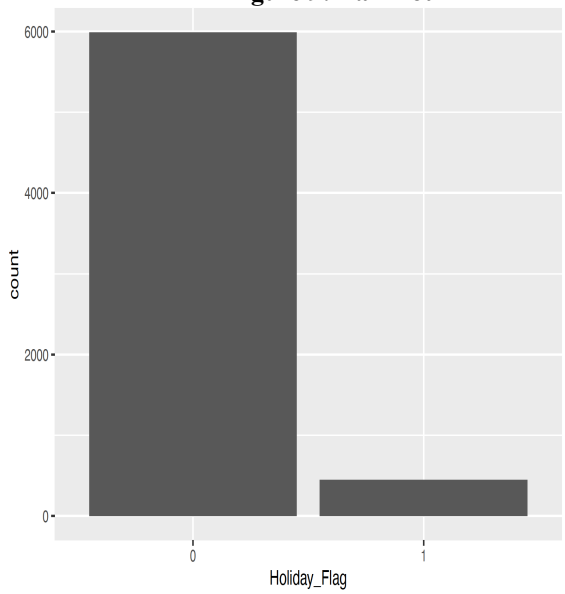
Addressing the Second Research Question (Using an Infer Package)

First, I grouped the data using the holiday flag variable. The holiday flag variable has two levels; 0 represents non-holiday week, 1 represents holiday week. Then, I calculated the mean and total counts (see Table 7) for each level of a holiday flag. I discovered that the non-holiday week has 5985 observations and the holiday week has 450 observations. After that, I used a bar plot (shown in Figure 9) to illustrate the data, each bar reflecting one of the holiday flag's factor levels. A bar plot also indicates an unbalanced design. As a result, it shows a true representation of the Walmart dataset. The observed test statistic is 81632.

Table 7: Tibble Table

Holiday_Flag	count	Mean
<fct>	<int>	<dbl>
0	5985	1041256.
1	450	1122888.

Figure 9: Bar Plot



Hypothesis Testing

I conducted a one-sided alternative hypothesis test where the null hypothesis and alternative hypothesis are as follows:
In words:
H_0: There is no difference in the means of the population which includes special holiday week and non-holiday week.
vs.
H_1: There is a significant difference in the means, where the special holiday week population is favored. In notations, where μ1 is the population of the special holiday week and μ2 is the population of the non-holiday week.

In notations:

$$H_0 : \mu_1 - \mu_2 = 0$$

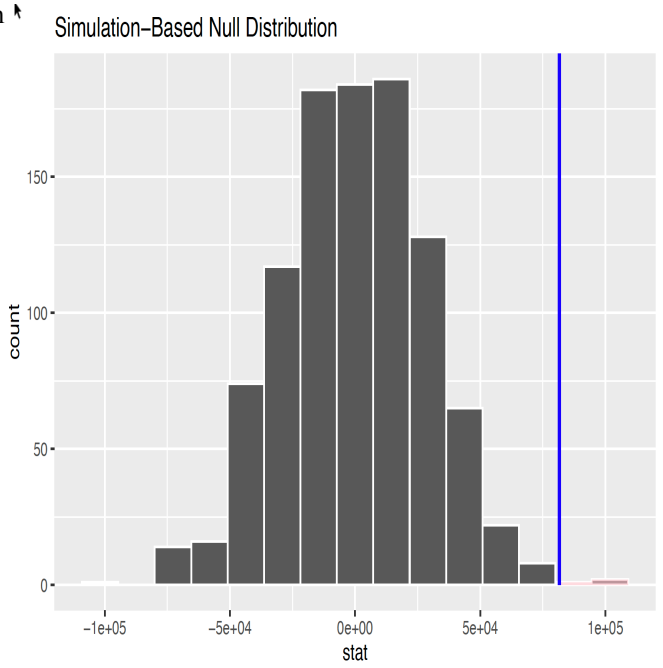
vs.

$$H_1 : \mu_1 - \mu_2 > 0$$

Visualized the Null Distribution

A histogram was used to show the null distribution and shaded p-value. I used the visualize verb from the infer package to visualize the null distribution (shown in Figure 10). After studying the null distribution and seeing that the observed difference in the sample means falls outside the middle 95% of the difference distribution, the one-sided test rejects the null hypothesis at a 5% α-level.

Figure 10: Null Distribution Plot



Decision

In Table 8, the p-value is determined to be 0.003, which is extremely small. I used the p-value approach to make the decision. Since the p-value is less than a commonly used αlpha value 0.05. As a result, I rejected the null hypothesis. I inferred that there is a significant difference in the means, where the special holiday week population is favored.

Table 8: P-value Table

p_value
<dbl>
0.003

Conclusion

The first inference drawn from the data is that weekly sales of the 45 Walmart stores are linked to the temperature, CPI, and unemployment. The multiple linear regression model proves the relationship between the variables by using weekly sales as the response and other economic factors such as temperature, CPI, unemployment as explanatory variables. The weekly sales of the 45 Walmart stores were negatively influenced by temperature, CPI, and unemployment. The second inference is drawn by comparing the bootstrapping regression methods with the traditional regression method. I found that the residual resampling technique works best in favor of the Walmart dataset. Also, the BCa 95% confidence interval is considered the best fit for the Walmart dataset. Subsequently, the third inference drawn started by conducting hypothesis testing using weekly sales as the response and the holiday flag as an explanatory variable. I rejected the null hypothesis using a computed p-value with an infer package. I found a significant difference in the means of the population, where the holiday week was favored. This project demonstrated that applying bootstrapping methods is very effective and provides very close results to the original coefficient estimates and is approachable. This project also ensures to the reader that the bootstrapping method is much easier to comprehend the coefficient results without being concerned about the LINE assumptions and extreme outliers.

References

- ***Walmart Dataset***
- ***Bootstrapping for Confidence Interval estimations and Hypothesis Testing***
- ***Github Link***

Appendices

Appendix: R Code

Load all of the essential libraries.

```{r}
# To load all of the packages in R, use the p_load function from
# the pacman library.
library(pacman)
p_load(car, tidyverse, MASS,bootstrap,boot, car,corrplot, infer,broom,purrr,
       rsample)
```

Load the dataset.

```{r}
# Create a variable called Walmart to store the dataset.
Walmart <- read_csv("/Users/sirpreetpadda/Desktop/Desktop stuff 2022/STAT 641/Stat 641 Final Project on Walmart/Walmart.csv")

# To retrieve the first four rows, use the head() function.
head(Walmart,4)

# To print the dataset's dimensions, use the dim() function.
dim(Walmart)
```

Data Preprocessing Step

```{r}
# To check for na values, use the is.na() function.
sum(is.na(Walmart))

# Build a pipeline with the Walmart dataset and store it in a
# new variable called Walmart.1.
Walmart.1 <- Walmart %>%
  dplyr::select(Weekly_Sales, Temperature, Fuel_Price, CPI, Unemployment) %>%
  drop_na()

# To observe the types of variables, use the glimpse() function.
glimpse(Walmart.1)
```

Fit a multiple linear regression model using all the predictors.

```{r}
# Create a fit variable to hold the multiple linear regression model.
```

```
fit <- lm(Weekly_Sales ~ ., data = Walmart.1)

# To print the summary statistics, use the summary() function.
summary(fit)
```
```{r}
# Use a backward stepwise selection technique with AIC.
Step <- step(fit, trace = FALSE)
coef(Step)

# Remove insignificant predictors from the multiple linear regression
# model, such as fuel price using an indexing position.
Walmart.2 <- Walmart.1[,-3]

# To retrieve the first six rows, use the head() function.
head(Walmart.2)
```
```{r}
# Determine the predictor-response correlations. Round up by 2 decimal places.
round(cor(Walmart.2),2)
```
Exploratory Data Analysis
```{r warning=FALSE}
# Using the corrplot package's corrplot() function.
# Make a visualization of the correlation matrix.
corrplot(cor(Walmart.2), method = "number", order = "hclust", col = 1:4,
         bg = "lightgrey")
```
```{r}
# Create a scatter plot of temperature versus weekly sales on a
# logarithmic scale using a Walmart dataset pipeline.
#The data points are represented by a holiday flag.
Walmart %>%
  mutate(Holiday_Flag = ifelse(Holiday_Flag == 1, "Holiday Week",
                               "Non-holiday Week"))%>%
  ggplot(aes(x = log(Temperature), y = log(Weekly_Sales))) +
  geom_point(aes(col = Holiday_Flag),alpha = 0.4) + geom_smooth(method = "lm",
                                                se = FALSE, col = "slategrey") +
  labs(x = "Temperature", y = "Weekly Sales",
       title = "Scatter plot of Weekly Sales Versus Temperature",
```

```
      col = "Holiday Flag")
```
```{r}
# Create a bar plot of holiday flags with two categories: holiday week
# and non-holiday week, relying on Walmart store weekly sales in billions.
# Make use of Walmart dataset pipeline.
Walmart %>%
  mutate(Holiday_Flag = factor(ifelse(Holiday_Flag == 1,
                                      "Holiday week", "Non-holiday Week"))) %>%
  group_by(Holiday_Flag) %>%
  ggplot(aes(y = Weekly_Sales/1000000000,x = Holiday_Flag)) +
  geom_bar(stat = "identity", aes(col = Holiday_Flag)) +
  labs(y = "Weekly Sales (In Billion USD)", x = "Holiday Flag",
       title = "Bar Plot of Weekly Sales of the Walmart Stores",
       col = "Holiday Flag ") +
  scale_y_continuous(breaks = seq(from = 0,to = 8, by = 1))
```
```{r warning=FALSE}
# To access the variables in the data, use the attach() function.
attach(Walmart.1)

# To see the relationship between variables, make a scatterplot matrix.
scatterplotMatrix(~Weekly_Sales + Temperature + Fuel_Price + CPI + Unemployment,
                  smooth = FALSE, cex = 0.2,regLine = list(col="green"),lty = 1)
```
Fit a multiple linear regression model using the significant variables.
```{r}
# Create a variable called fit2 that holds the multiple linear regression
# model with selected variables.
fit2 <- lm(Weekly_Sales ~ Temperature + CPI + Unemployment, data = Walmart.2)

# Using the summary() function, print the summary statistics.
summary(fit2)
```
```{r}
# Diagnostic Plots
par(mfrow = c(2,2))
# With the plot() function, examine the diagnostic plots.
plot(fit2, which = 1:4)
```

```
# Find the outliers length.
ind <- length(which(abs(rstandard(fit2))>2))
ind
```
Check for multicollinearity Issues.
```{r}
# Using the variance inflation factor VIF(), check for multicollinearity Issues.
# Round up by 2 decimal places.
VIF <- round(vif(fit2),2)

# Print the results.
print(ifelse(VIF > 5, "Multicollinearity Issue",VIF))
```
Bootstrapping Methods
Apply the Observation resampling / random x resampling method.
```{r}
# Set the seed to reproduce the results.
set.seed(222)

# Create a function named boot_c that takes data and i (indices) as parameters.
# Fit a multiple linear regression model using ith indices.
# The function boot_c returns the coefficients of the multiple
# regression linear model.
boot_c <- function(data, i){
  dat <- data[i,]
  # Fit a multiple linear regression model.
  mod <- lm(Weekly_Sales ~ Temperature + CPI + Unemployment, data = dat)
  # Return coefficient vector using the coef() function.
  return(coef(mod))
}

# Create a variable named boot.random that calls the boot() function with
# certain arguments like data, statistics of interest, and R
# (number of bootstrap samples).
boot.random <- boot(data = Walmart.2, statistic = boot_c, R = 8000)

# To print the results, call boot.random variable.
boot.random
```
Visualize the bootstrap replicates.

```{r}
# To check the diagnostic plots, use the plot() function.
plot(boot.random, index = 1) # index = 1 represents intercept (beta_0 hat)
plot(boot.random, index = 2) # index = 2 represents beta_1 hat
plot(boot.random, index = 3) # index = 3 represents beta_2 hat
plot(boot.random, index = 4) # index = 4 represents beta_3 hat
```
```{r}
# 95% confidence interval
confint(fit2)
```
```{r}
# Using the boot.ci() function, construct a 95 percent bootstrap confidence
# interval. Pass arguments such as boot.random, indexing position, confidence
# level, and type ("norm","perc","bca").
boot.ci(boot.random, index = 2, conf = 0.95, type = c("norm","perc","bca"))
boot.ci(boot.random, index = 3, conf = 0.95, type = c("norm","perc","bca"))
```
Apply the residual resampling / fixed x resampling method.
```{r}
# Set the seed to reproduce the results.
set.seed(222)

# The fitted multiple linear regression model is stored in the fit2 variable.
# To extract the fitted values from fit2, use the fitted() function.
# Save the results to a new variable named fitt.
fitt <- fitted(fit2)

# Similarly use the residuals() function to extract the residual
# values from fit2.
e <- residuals(fit2)

# Using fit2, create a design matrix X.
X <- model.matrix(fit2)

# Make a model-holding function called boot fun fixed. Pass some arguments,
# such as data and i. (indices). Returns the coefficients.
boot_fun_fixed <- function(data, i){
  y_b <- fitt + e[i]
  # Fit a model excluding Intercept.
```

```
  mod <- lm(y_b ~ X - 1)
  return(coef(mod))
}

# Create a variable named boot.fixed that calls the boot() function
# with certain arguments like data, statistics of interest, and R
# (number of bootstrap samples).
boot.fixed <- boot(Walmart.2, boot_fun_fixed, R = 8000)

# To print the results, call boot.fixed variable.
boot.fixed
```
Visualize the bootstrap replicates.
```{r}
# To check the diagnostic plots, use the plot() function.
plot(boot.fixed, index = 1)
plot(boot.fixed, index = 2)
plot(boot.fixed, index = 3)
plot(boot.fixed, index = 4)
```
```{r}
# Using the boot.ci() function, construct a 95 percent bootstrap confidence
# interval. Pass arguments such as boot.fixed, indexing position, confidence
# level, and type ("norm","perc","bca").
boot.ci(boot.fixed, index = 2, conf = 0.95, type = c("norm","perc","bca"))
boot.ci(boot.fixed, index = 3, conf = 0.95, type = c("norm","perc","bca"))
```
```{r}
# Add smooth density curve plots with indexing position 2
# using the plot() function.
par(mfrow = c(1,2), mar = c(2,2,2,2))
plot(density(boot.random$t[,2]), main = "Density Plot (x is random,i=2)")# i is an indexing position.
plot(density(boot.fixed$t[,2]),main = "Density Plot (x is fixed,i=2)")# i is an indexing position.
```
Address the second research question.
```{r}
# Convert a variable called holiday flag to a factor using the
# Walmart1 pipeline. Store into a new variable called Walmart2.
Walmart.3 <- Walmart %>%
  dplyr::select(Weekly_Sales, Holiday_Flag) %>%
```

```
  # Use mutate () function from the dplyr package to alter
  # holiday flag variable.
  mutate(Holiday_Flag = factor(Holiday_Flag))
```
```{r}
# To incorporate the pipeline, create a new variable called Walmart2.
Walmart.3 %>%
  # Use group_by() function to perform data operations on groups.
  # Pass an argument holiday flag.
  group_by(Holiday_Flag) %>%
  # On the grouped data by holiday flag, use the summarise() function.
  # Compute the mean of each group.
  summarise(
    count = n(),
    Mean = round(mean(Weekly_Sales),2)
  )
# Using the ggplot2 approach, visualize the bar plot.
Walmart.3 %>%
  ggplot(aes(x = Holiday_Flag,fill = Weekly_Sales)) + geom_bar() +
  labs(fill = "Holiday Flag")

# To examine the levels of each category of holiday flag,
# use the table() function.
table(Walmart.3$Holiday_Flag)
```
Compute the Observed Test Statistic
```{r}
# Create a variable called obs_test_stat to contain the
# Walmart2 dataset's pipeline.
obs_test_stat <- Walmart.3 %>%
  # Use a specify() function to specify the variables of interest.
  specify(Weekly_Sales ~ Holiday_Flag) %>%

  # Use a calculate() function to compute the observed test statistics.
  calculate(stat = "diff in means", order = c(1,0))

# Round up the observed test statistic by 2 decimal places.
round(obs_test_stat,2)

# To double-check, I manually computed observed test statistics.
```

```
# Store the results into a variable called obs_diff_mean.
obs_diff_mean = 1122888-1041256
obs_diff_mean
```
Null distribution
Under null distribution I assume the null hypothesis is true.
```{r}
# Create a null_dist variable that carries the Walmart2 pipeline.
null_dist <- Walmart.3 %>%
  # Use a specify() function to specify the variables of interest.
  specify(Weekly_Sales ~ Holiday_Flag) %>%

  # To set the null equals independence, use the hypothesis() function.
  hypothesize(null = "independence") %>%

  # Use the generate () function to generate bootstrap replicates using a
  # permutation resampling technique.
  generate(reps = 1000, type = "permute") %>%

  # Use the calculate() function and pass in some arguments like
  # "difference in means" statistics of interest and order levels.
  calculate(stat = "diff in means", order = c(1,0))

# To retrieve the first five rows of the bootstrap replicates,
# use the head() function.
head(null_dist,5)
```
Visualize the null distribution.
```{r}
# Visualize the null distribution using the visualize() function.
null_dist %>%
  # Visualize the null distribution using the visualize() function.
  visualize() +
  # To display the observed test statistics and p-value in the shaded region,
  # use the shade_p_value() function.
  shade_p_value(obs_stat = obs_test_stat, direction = "right", col = "blue",
                lty = 1, lwd = 1)
```
Compute the achieved significance level (ASL) / p-value.
```{r}


```

```
# By passing parameters like the observed test statistic and direction to
# the get p value() function, one can compute the p-value.
null_dist %>%
  get_p_value(obs_stat = obs_test_stat, direction = "right")
```