Recycling project code

Aayush

2024-06-08

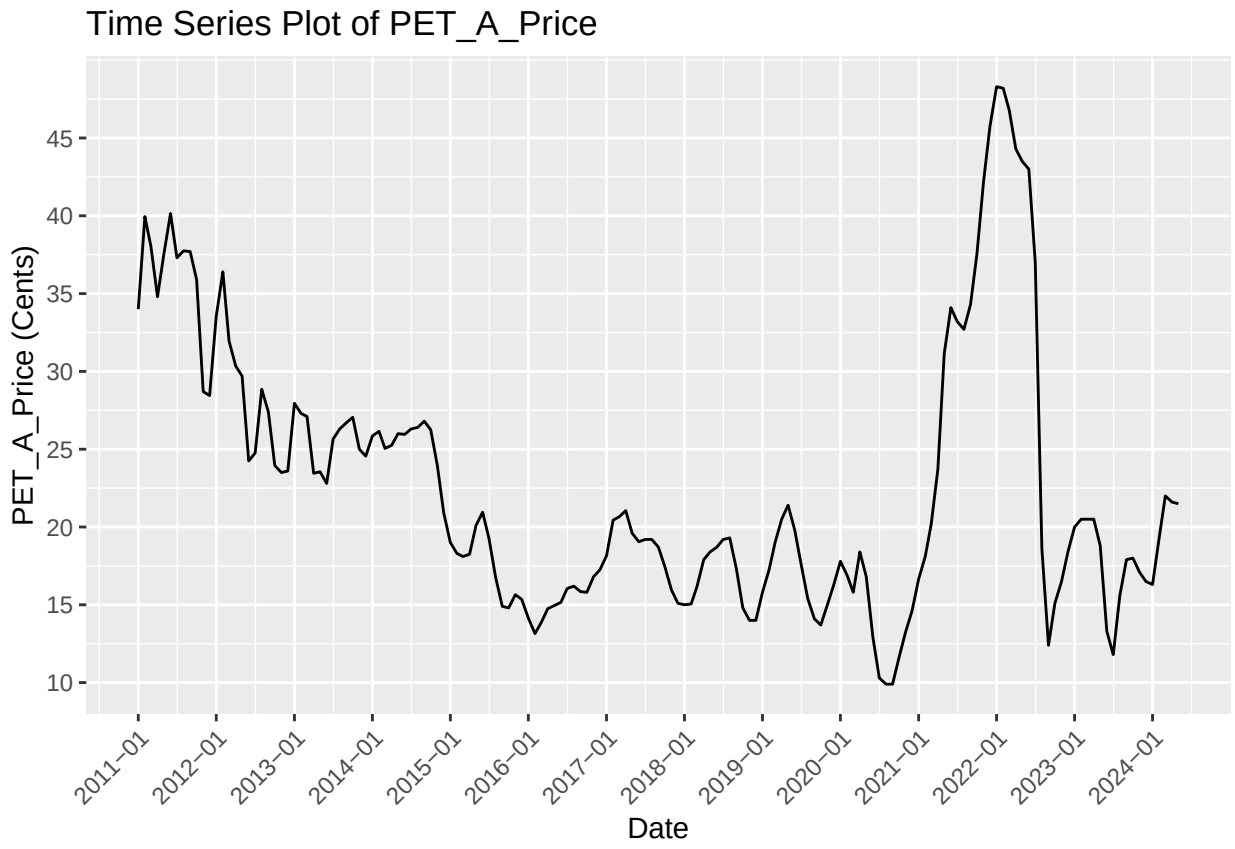
```
library(pacman)  # Load necessary libraries
p_load(readr, tidyverse, lubridate, jtools)

# Load PRCC Bid Pricing data
Bid_Pricing <- read.csv("C:\\Users\\Aayush\\Documents\\California recycling project\\PRCC Bid Pricing.cs

# Load PRCC Volume data
Volume <- read.csv("C:\\Users\\Aayush\\Documents\\California recycling project\\PRCC Volume.csv")
```

```
# Convert 'Date' column to date format
Bid_Pricing$Date <- my(Bid_Pricing$Date)

# Add new columns with computed prices and ratios
Bid_Pricing <- Bid_Pricing %>%
  mutate(PET_A_Price = (PET.A.LBC + PET.A.OAK) / 2,
         PET_B_Price = (PET.B.LBC + PET.B.OAK) / 2,
         Ratio_B_A = PET_B_Price / PET_A_Price,
         BWTF_Price = (BwTF.LBC + BwTF.OAK) / 2) %>%
  rename(PET_Price = PET.TF) %>%
  mutate(PET_A_Price = PET_A_Price * 100,
         PET_B_Price = PET_B_Price * 100,
         BWTF_Price = BWTF_Price * 100,
         PET_Price = PET_Price * 100)
```

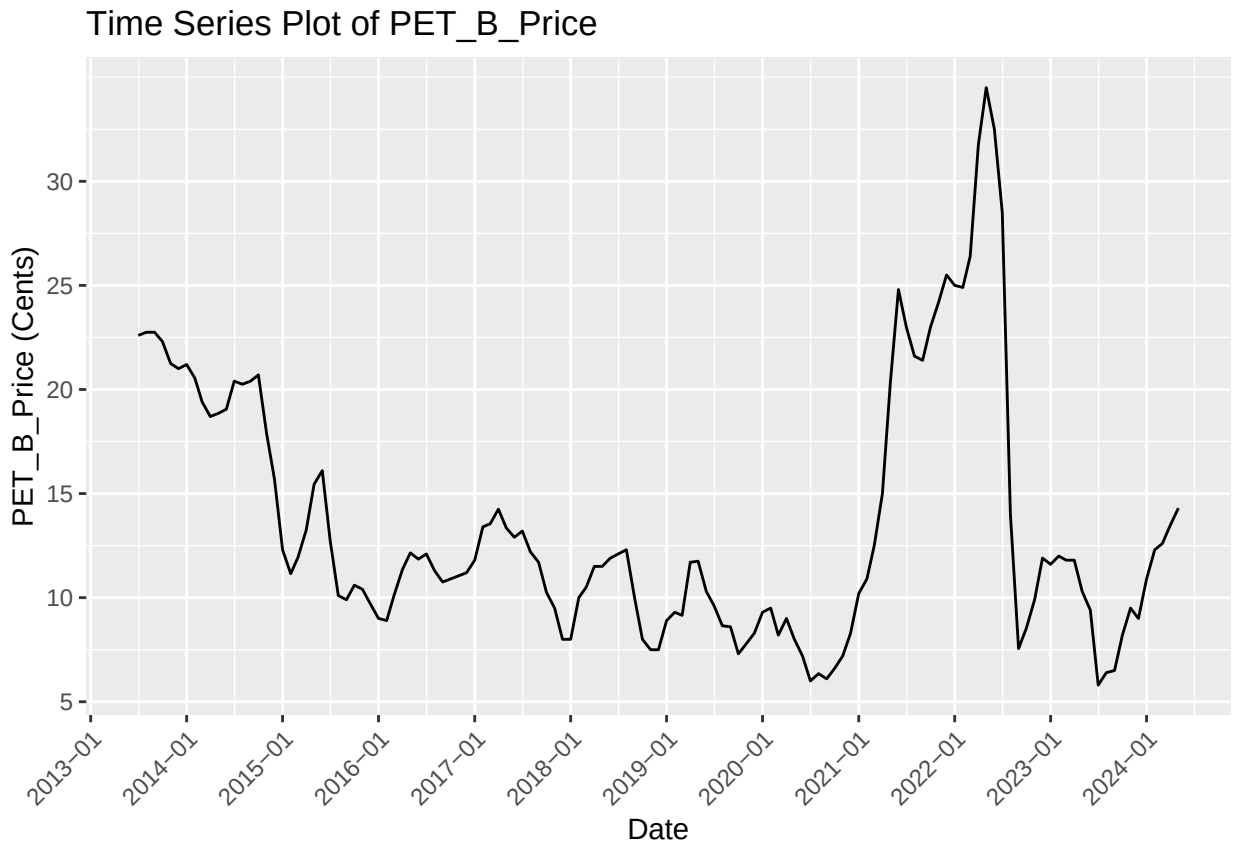
```
# Convert 'mo' and 'yr' to 'date_column' and reshape data
Volume <- Volume %>%
  mutate(date_column = as.Date(paste(yr, mo, "01", sep = "-"))) %>%
  group_by(date_column) %>%
  mutate(column_name = case_when(
    material %in% c("A", "PET A") ~ "PET_A_lbs",
    material %in% c("B", "PET B") ~ "PET_B_lbs",
    material == "PET BwTF" ~ "PET_BwTF_lbs",
    material == "PET TF" ~ "PET_TF_lbs"
  )) %>%
  select(date_column, column_name, lbs) %>%
  pivot_wider(names_from = column_name, values_from = lbs)
```

```
plot_path <- "C:\\Users\\Aayush\\Documents\\California recycling project\\plots"

# Time series plot for PET_A_Price
time_series_plot_PET_A_Price <- ggplot(Bid_Pricing, aes(x = Date, y = PET_A_Price)) +
```

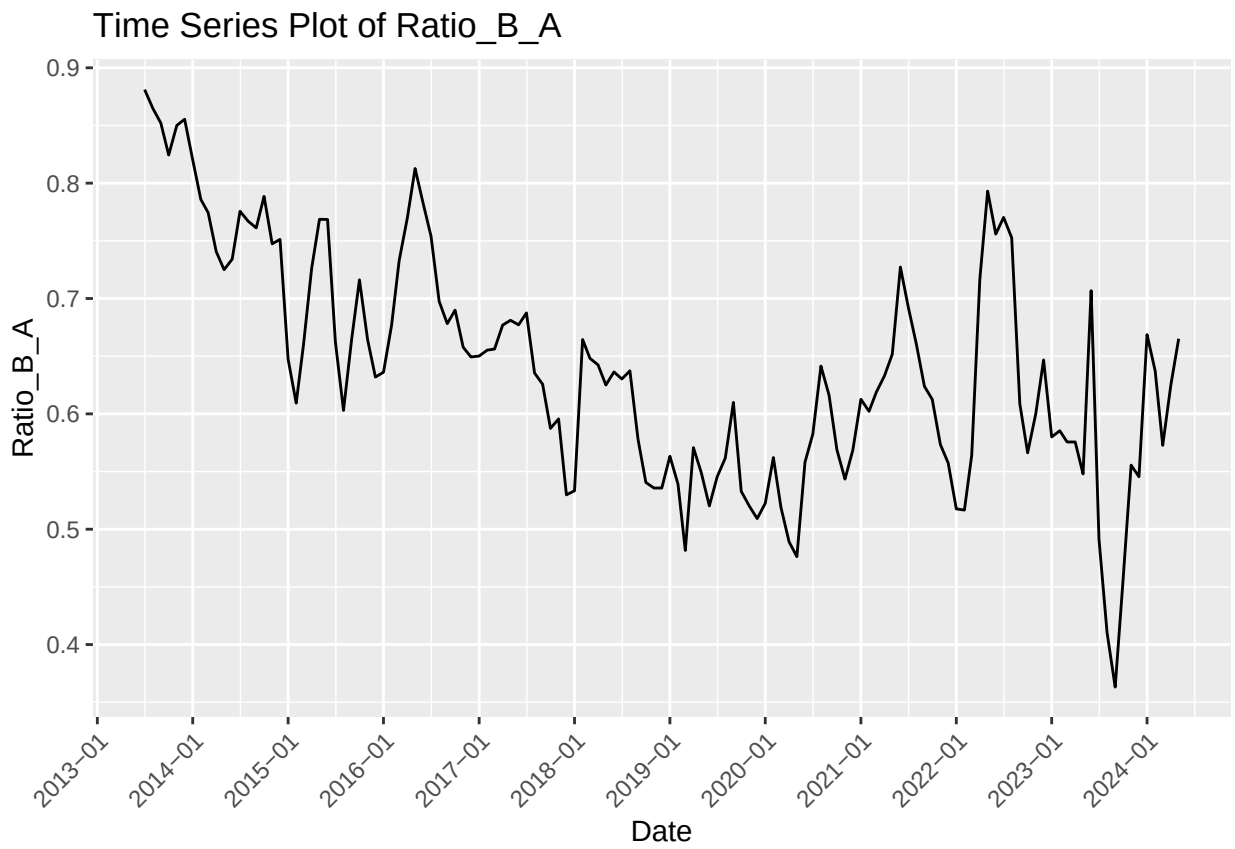
```
  geom_line() +
  labs(x = "Date", y = "PET_A_Price (Cents)", title = "Time Series Plot of PET_A_Price") +
  scale_x_date(date_labels = "%Y-%m", date_breaks = "1 year") +
  scale_y_continuous(breaks = seq(0, max(Bid_Pricing$PET_A_Price, na.rm = TRUE), by = 5)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

print(time_series_plot_PET_A_Price)
```



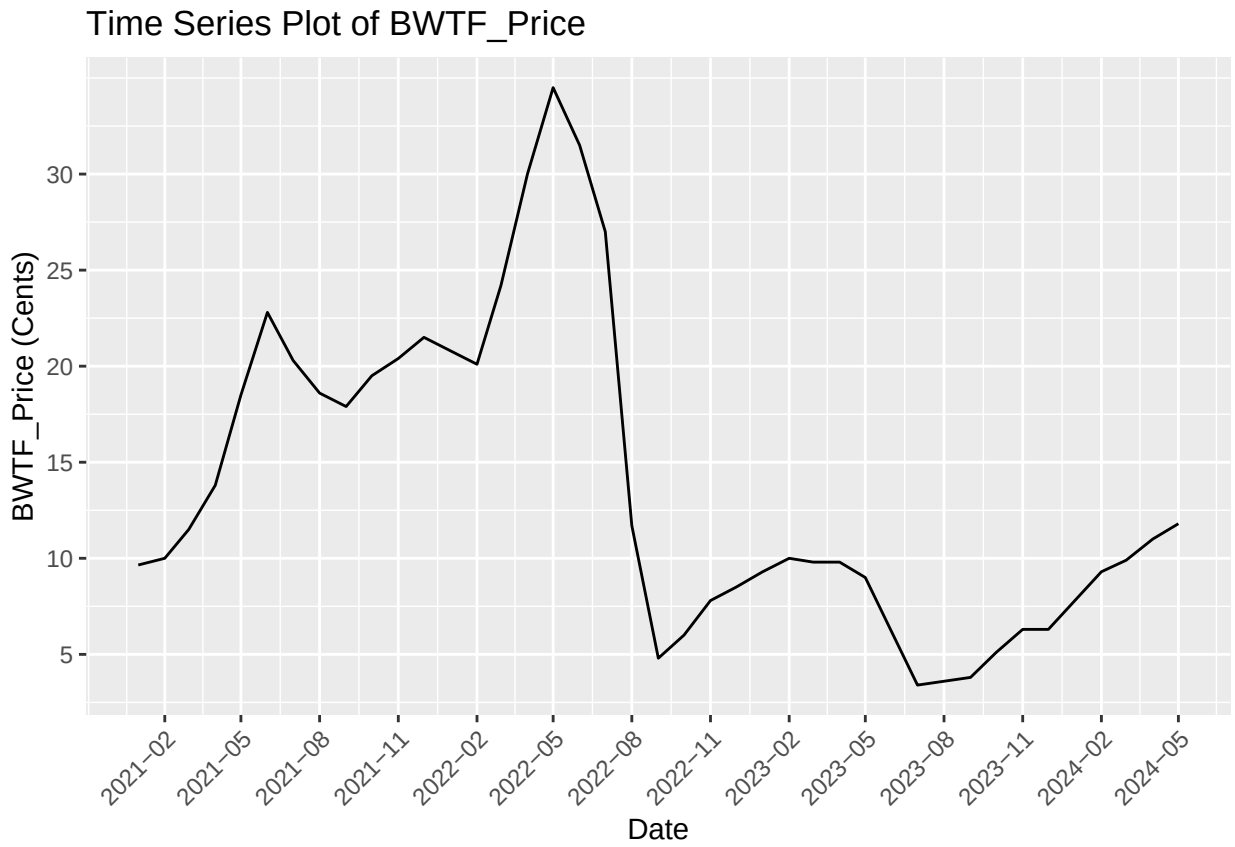

```
ggsave(filename = file.path(plot_path, "time_series_plot_PET_A_Price.png"), plot = time_series_plot_PET

# Filter data from July 2013 and plot PET_B_Price
Bid_Pricing_filtered_B <- Bid_Pricing %>%
  filter(Date > as.Date("2013-06-30"))

plot_B <- ggplot(Bid_Pricing_filtered_B, aes(x = Date, y = PET_B_Price)) +
  geom_line() +
  labs(x = "Date", y = "PET_B_Price (Cents)", title = "Time Series Plot of PET_B_Price") +
  scale_x_date(date_labels = "%Y-%m", date_breaks = "1 year") +
  scale_y_continuous(breaks = seq(0, max(Bid_Pricing_filtered_B$PET_B_Price, na.rm = TRUE), by = 5)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

ggsave(filename = file.path(plot_path, "time_series_plot_PET_B_Price.png"), plot = plot_B, width = 10,
print(plot_B)
```



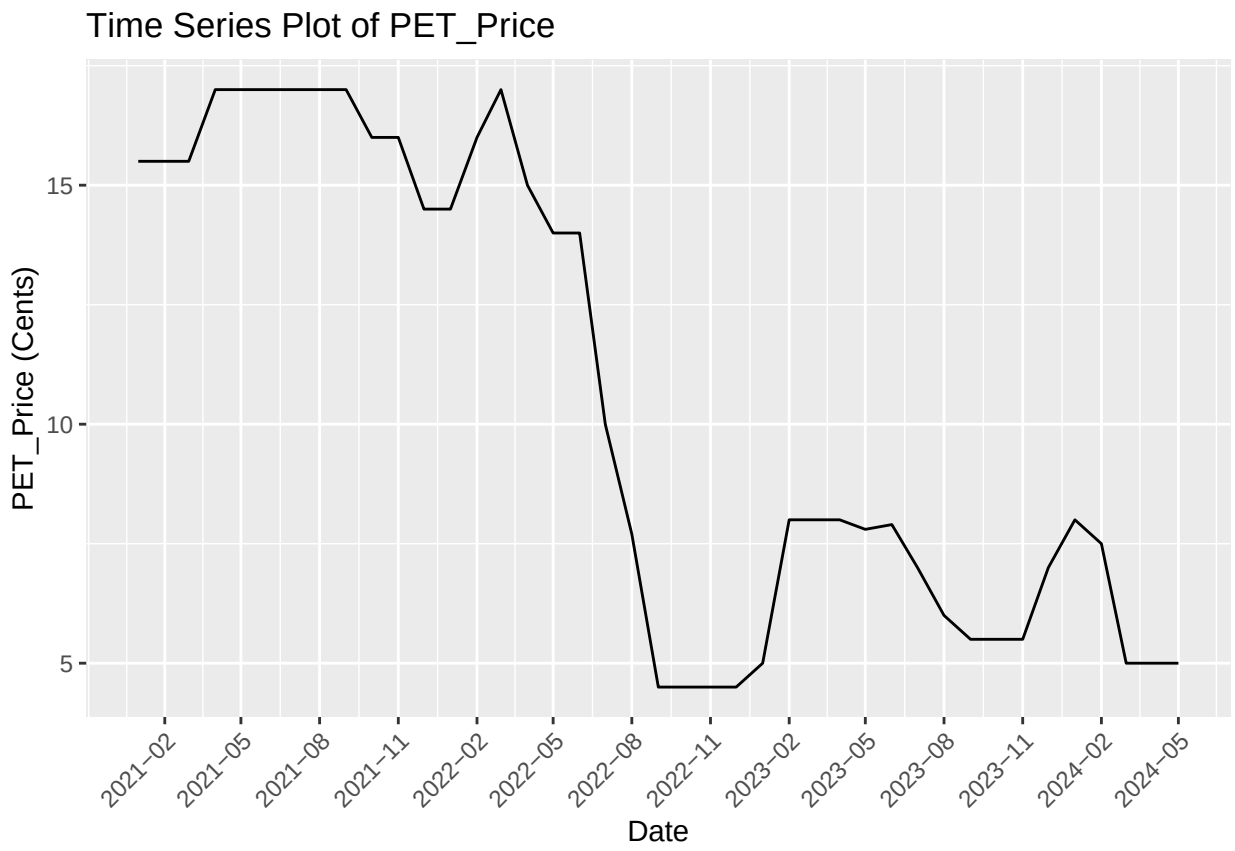

```r
# Plot Ratio_B_A
plot_ratio <- ggplot(Bid_Pricing_filtered_B, aes(x = Date, y = Ratio_B_A)) +
  geom_line() +
  labs(x = "Date", y = "Ratio_B_A", title = "Time Series Plot of Ratio_B_A") +
  scale_x_date(date_labels = "%Y-%m", date_breaks = "1 year") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

ggsave(filename = file.path(plot_path, "time_series_plot_Ratio_B_A.png"), plot = plot_ratio, width = 10
print(plot_ratio)
```



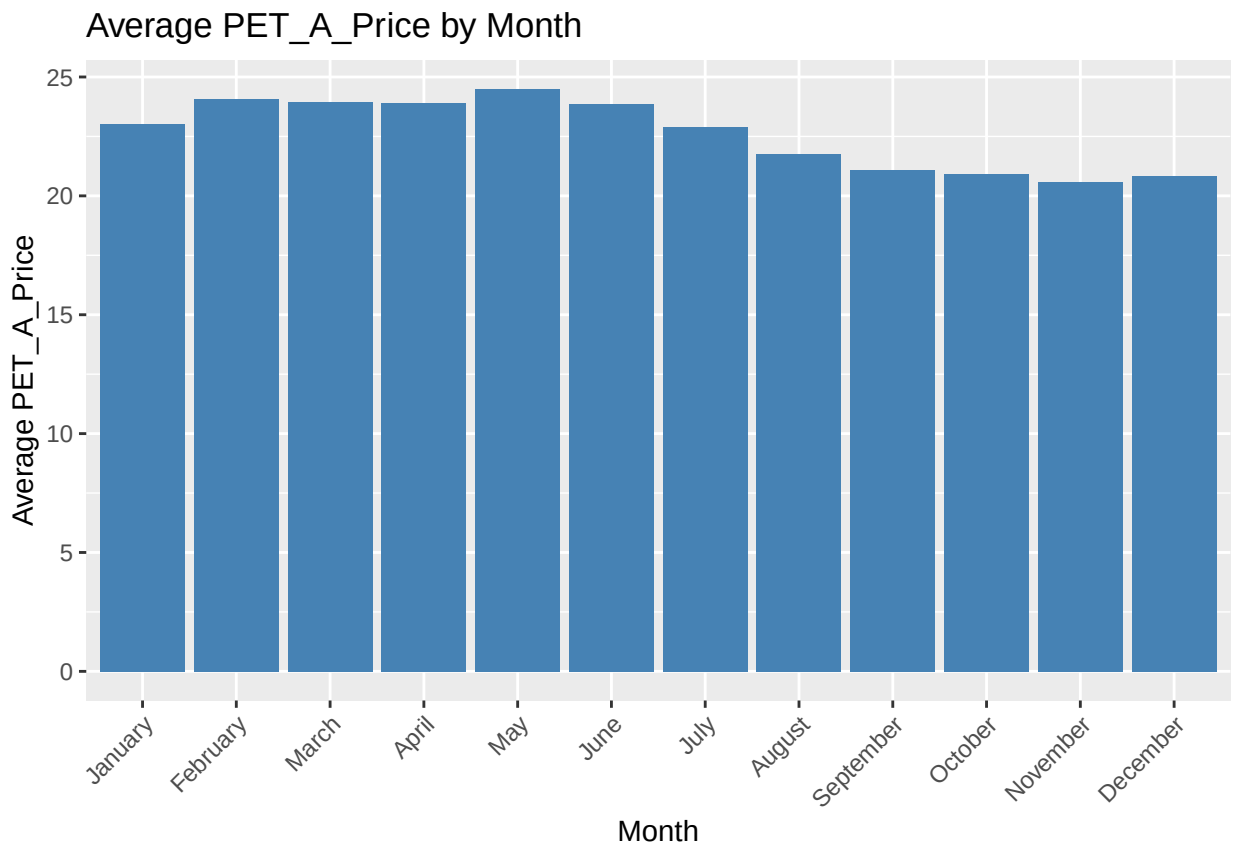

```
# Filter data from January 2021 and plot BWTF_Price
Bid_Pricing_filtered <- Bid_Pricing %>%
  filter(Date >= as.Date("2021-01-01"))

plot_bwtf_price <- ggplot(Bid_Pricing_filtered, aes(x = Date, y = BWTF_Price)) +
  geom_line() +
  labs(x = "Date", y = "BWTF_Price (Cents)", title = "Time Series Plot of BWTF_Price") +
  scale_x_date(date_labels = "%Y-%m", date_breaks = "3 months") +
  scale_y_continuous(breaks = seq(0, max(Bid_Pricing_filtered$BWTF_Price, na.rm = TRUE), by = 5)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

ggsave(filename = file.path(plot_path, "time_series_plot_BWTF_Price.png"), plot = plot_bwtf_price, width
print(plot_bwtf_price)
```




```r
# Plot PET_Price
plot_pet_price <- ggplot(Bid_Pricing_filtered, aes(x = Date, y = PET_Price)) +
  geom_line() +
  labs(x = "Date", y = "PET_Price (Cents)", title = "Time Series Plot of PET_Price") +
  scale_x_date(date_labels = "%Y-%m", date_breaks = "3 months") +
  scale_y_continuous(breaks = seq(0, max(Bid_Pricing_filtered$PET_Price, na.rm = TRUE), by = 5)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

ggsave(filename = file.path(plot_path, "time_series_plot_PET_Price.png"), plot = plot_pet_price, width =
print(plot_pet_price)
```

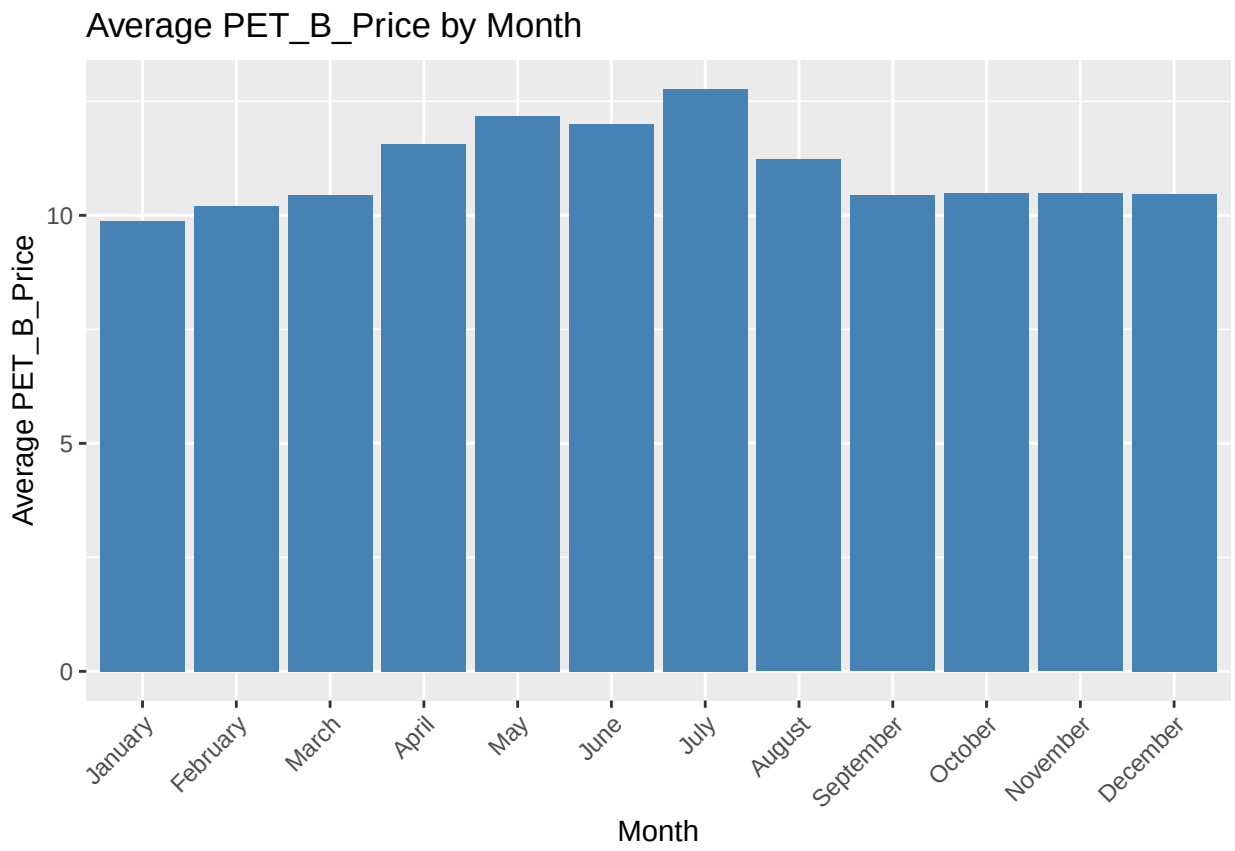



```
plot_path <- "C:\\Users\\Aayush\\Documents\\California recycling project\\plots"



# Compute the average PET_A_Price for each month
average_price_by_month_PET_A <- Bid_Pricing %>%
  mutate(Month = format(as.Date(Date), "%B")) %>%  # Extract month names
  group_by(Month) %>%
  summarize(Average_PET_A_Price = mean(PET_A_Price, na.rm = TRUE)) %>%
  ungroup() %>%
  mutate(Month = factor(Month, levels = month.name))  # Ensure months are in order

# Create the bar graph
bar_plot_PET_A <- ggplot(average_price_by_month_PET_A, aes(x = Month, y = Average_PET_A_Price)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(x = "Month", y = "Average PET_A_Price", title = "Average PET_A_Price by Month") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Display the plot
print(bar_plot_PET_A)
```



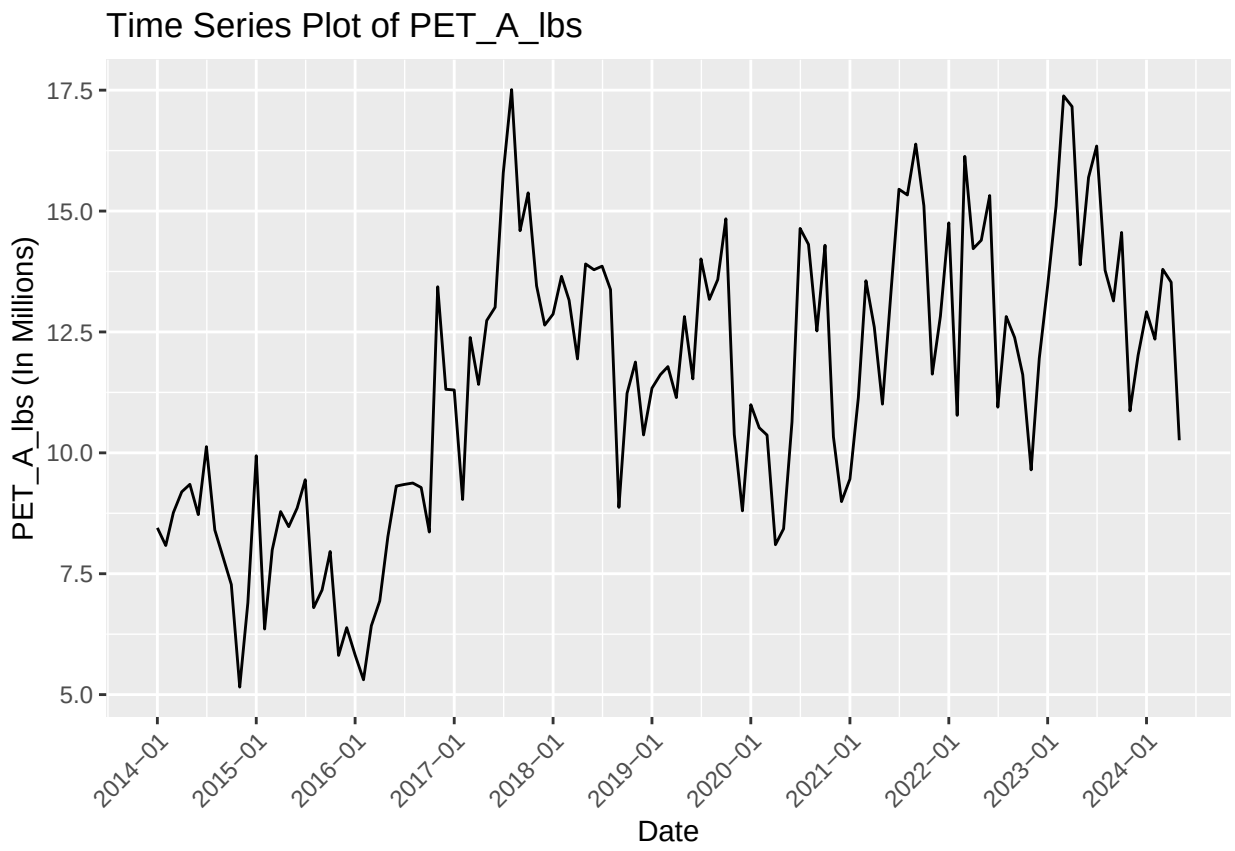

```
# Save the plot
ggsave(filename = file.path(plot_path, "bar_plot_average_monthly_PET_A.png"), plot = bar_plot_PET_A, wi
```

```
# Compute the average PET_B_Price for each month
average_price_by_month_PET_B <- Bid_Pricing %>%
  mutate(Month = format(as.Date(Date), "%B")) %>%  # Extract month names
  group_by(Month) %>%
  summarize(Average_PET_B_Price = mean(PET_B_Price, na.rm = TRUE)) %>%
  ungroup() %>%
  mutate(Month = factor(Month, levels = month.name))  # Ensure months are in order

# Create the bar graph
bar_plot_PET_B <- ggplot(average_price_by_month_PET_B, aes(x = Month, y = Average_PET_B_Price)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(x = "Month", y = "Average PET_B_Price", title = "Average PET_B_Price by Month") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Display the plot
print(bar_plot_PET_B)
```



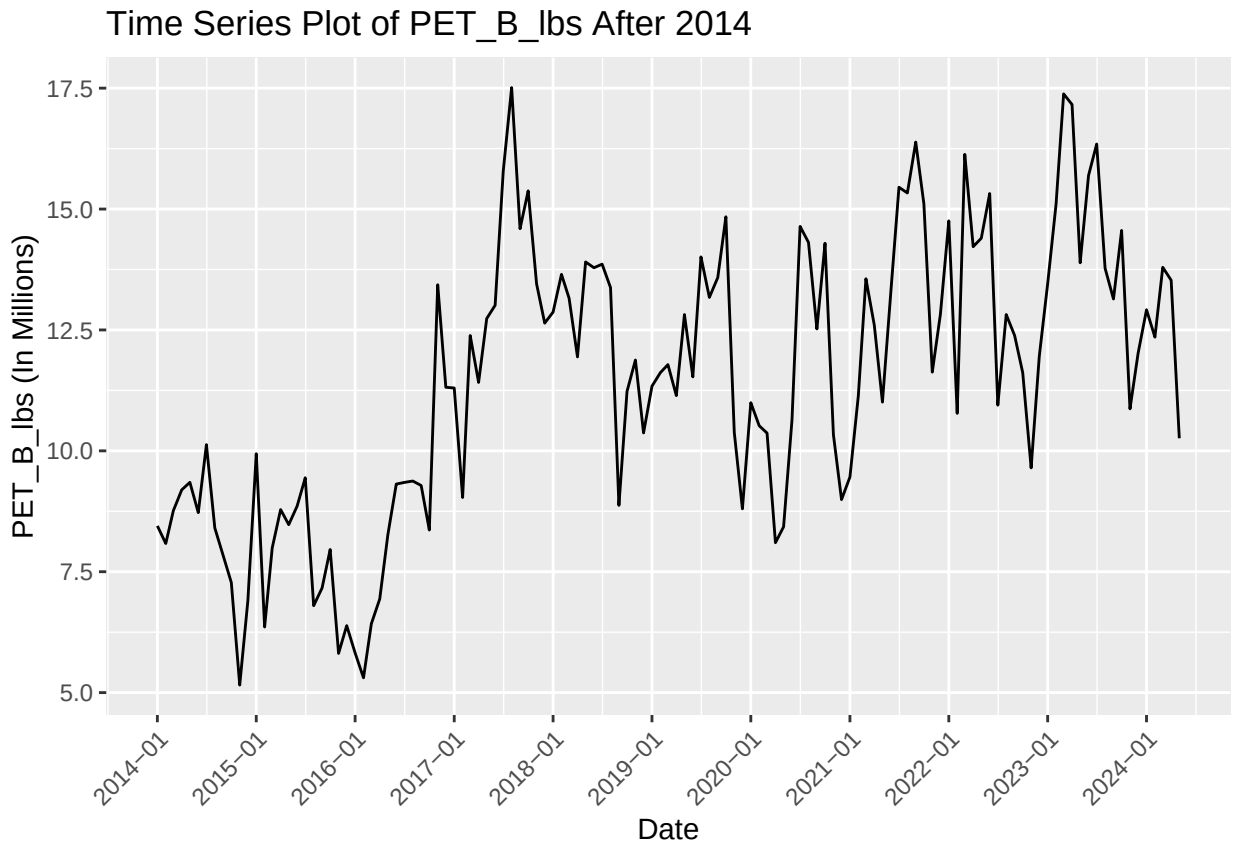

```
# Save the plot
ggsave(filename = file.path(plot_path, "bar_plot_average_monthly_PET_B.png"), plot = bar_plot_PET_B, wi
```

```
# Define the path to save the plot
plot_path <- "C:\\Users\\Aayush\\Documents\\California recycling project\\plots"

# Filter data to start from January 2014
Volume_Filtered_after_2014 <- Volume %>%
  filter(date_column >= as.Date("2014-01-01"))

# Create the time series plot for PET_A_lbs
time_series_plot_PET_A_lbs <- ggplot(Volume_Filtered_after_2014, aes(x = date_column, y = as.integer(PE
  geom_line() +
  labs(x = "Date", y = "PET_A_lbs (In Millions)", title = "Time Series Plot of PET_A_lbs") +
  scale_x_date(date_labels = "%Y-%m", date_breaks = "1 year") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))


# Display the plot
print(time_series_plot_PET_A_lbs)
```



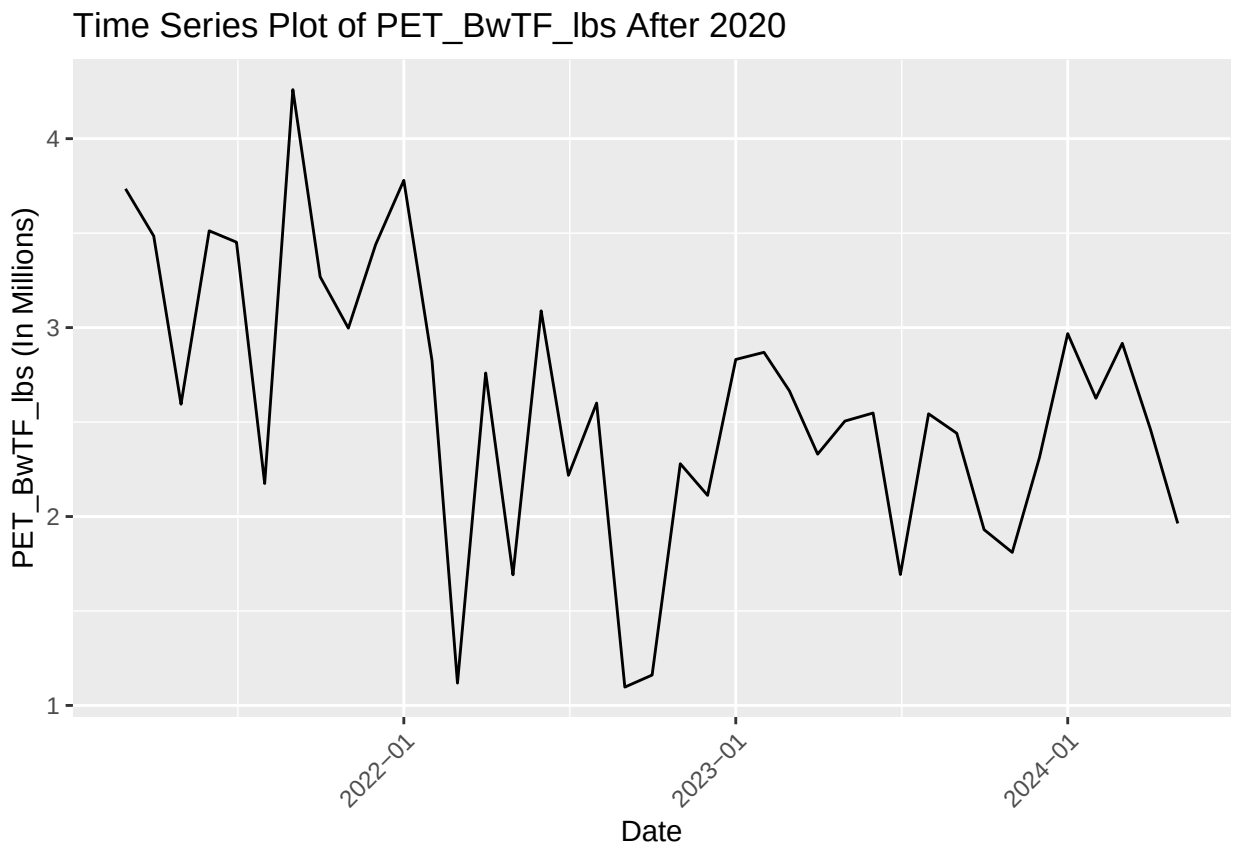

```
# Save the plot
ggsave(filename = file.path(plot_path, "time_series_plot_PET_A_lbs.png"), plot = time_series_plot_PET_A
```

```
# Create the time series plot for PET_B_lbs
time_series_plot_PET_B_lbs_after_2014 <- ggplot(Volume_Filtered_after_2014, aes(x = date_column, y = as
  geom_line() +
  labs(x = "Date", y = "PET_B_lbs (In Millions)", title = "Time Series Plot of PET_B_lbs After 2014") +
  scale_x_date(date_labels = "%Y-%m", date_breaks = "1 year") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Display the plot
print(time_series_plot_PET_B_lbs_after_2014)
```



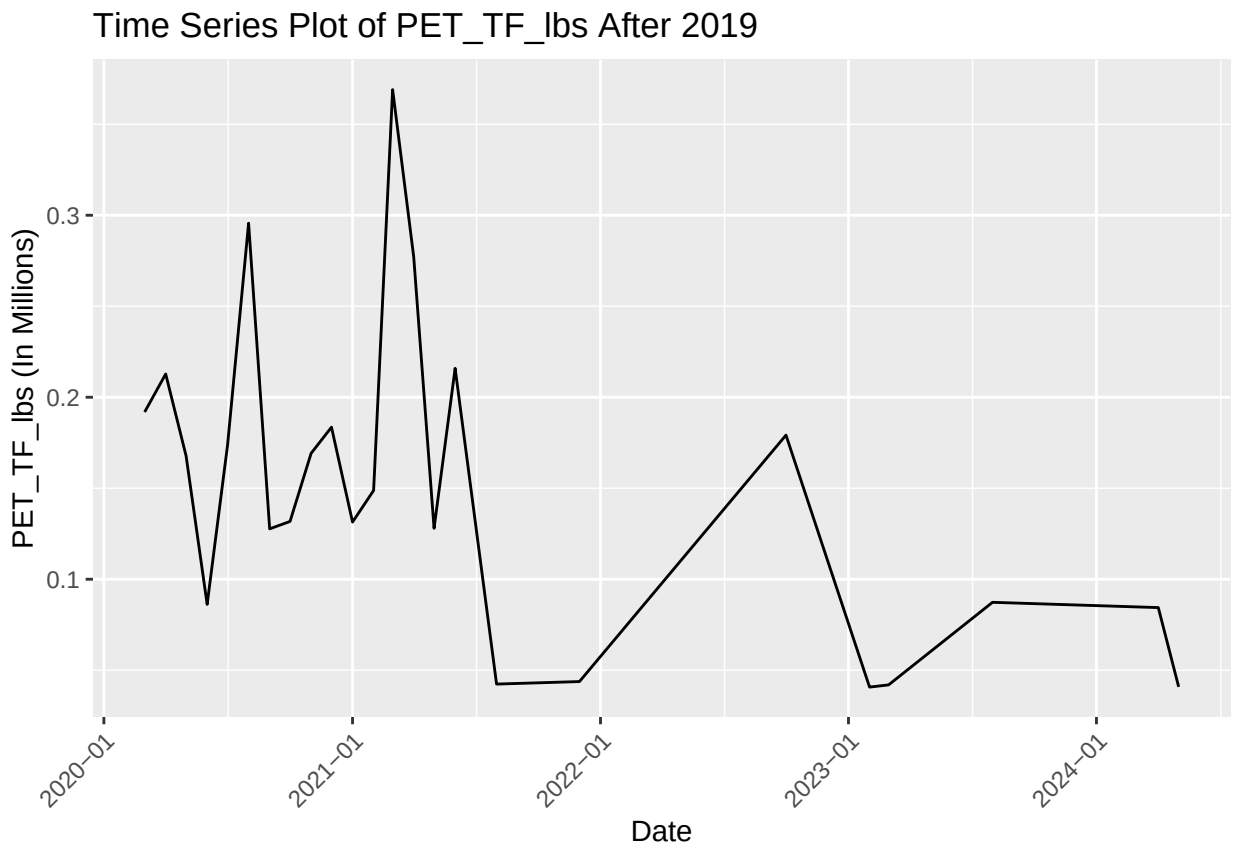

```r
# Save the plot
ggsave(filename = file.path(plot_path, "time_series_plot_PET_B_lbs.png"), plot = time_series_plot_PET_B_
```

```r
# Filter data after 2020
Volume_Filtered_after_2020 <- Volume %>%
  filter(date_column >= as.Date("2021-03-01")) %>%
  filter(!is.na(PET_BwTF_lbs))  # Remove rows with NA in PET_BwTF_lbs

# Create the time series plot for PET_BwTF_lbs (log-transformed)
time_series_plot_PET_BwTF_lbs_after_2020 <- ggplot(Volume_Filtered_after_2020, aes(x = date_column, y =
  geom_line() +
  labs(x = "Date", y = "PET_BwTF_lbs (In Millions)", title = "Time Series Plot of PET_BwTF_lbs After 20
  scale_x_date(date_labels = "%Y-%m", date_breaks = "1 year") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Display the plot
print(time_series_plot_PET_BwTF_lbs_after_2020)
```




```
# Save the plot
ggsave(filename = file.path(plot_path, "time_series_plot_PET_BwTF_lbs_after_2020.png"), plot = time_seri
```

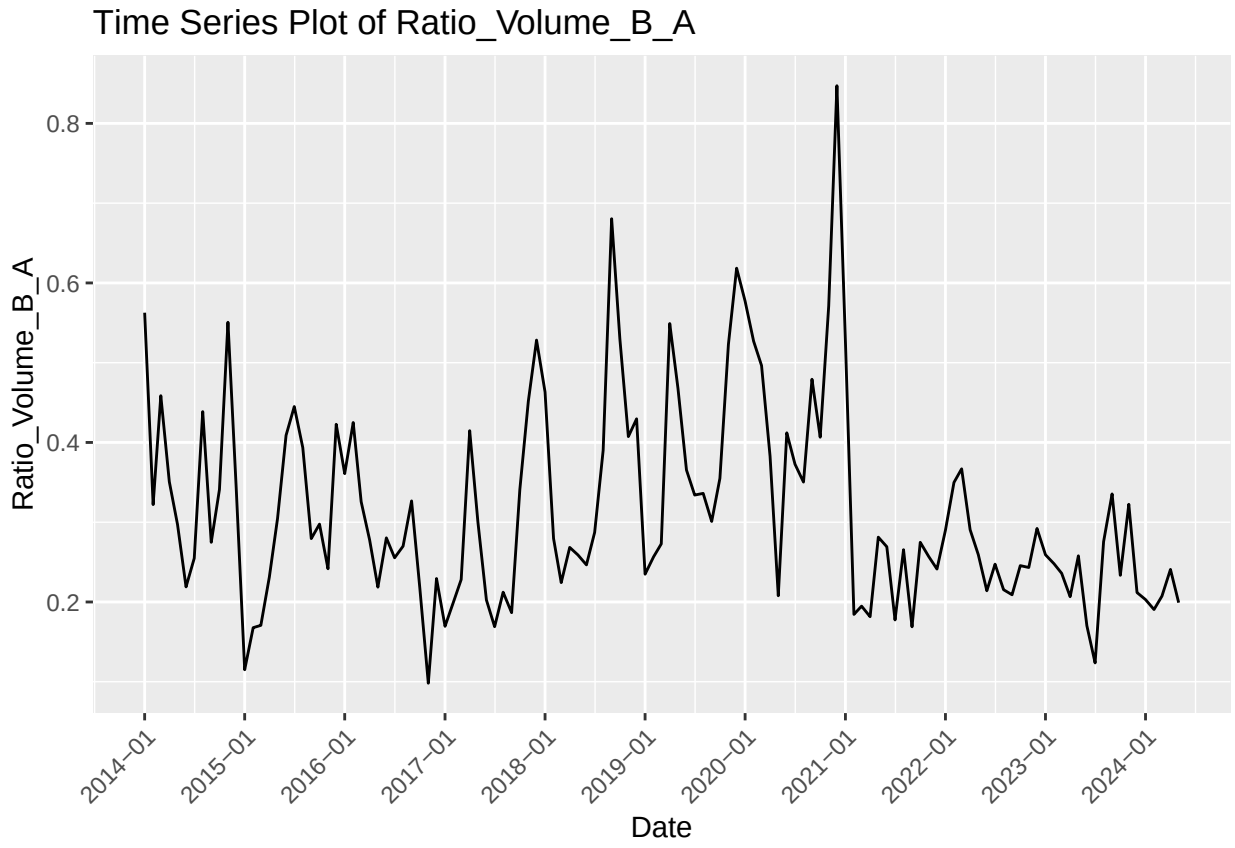
```
# Filter data after January 1, 2019
Volume_Filtered_after_2019 <- Volume %>%
  filter(date_column >= as.Date("2019-01-01")) %>%
  filter(!is.na(PET_TF_lbs))  # Remove rows with NA in PET_TF_lbs

# Create the time series plot for PET_TF_lbs
time_series_plot_PET_TF_lbs_after_2019 <- ggplot(Volume_Filtered_after_2019, aes(x = date_column, y = as
  geom_line() +
  labs(x = "Date", y = "PET_TF_lbs (In Millions)", title = "Time Series Plot of PET_TF_lbs After 2019")
  scale_x_date(date_labels = "%Y-%m", date_breaks = "1 year") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Display the plot
print(time_series_plot_PET_TF_lbs_after_2019)
```



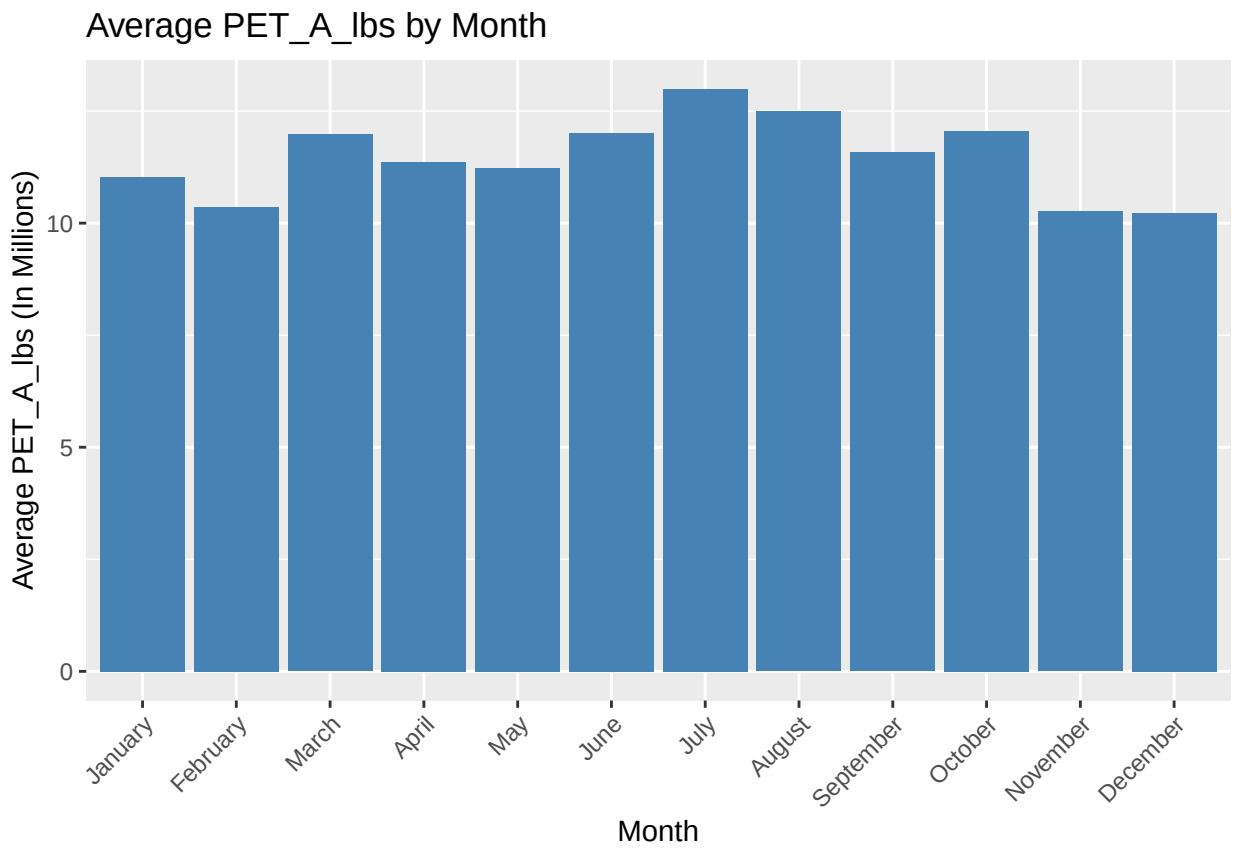

```
# Save the plot
ggsave(filename = file.path(plot_path, "time_series_plot_PET_TF_lbs_after_2019.png"), plot = time_series
```

```
# Convert columns to numeric if they are not already
Volume <- Volume %>%
  mutate(PET_B_lbs = as.numeric(PET_B_lbs),
         PET_A_lbs = as.numeric(PET_A_lbs)) %>%
  mutate(Ratio_Volume_B_A = PET_B_lbs / PET_A_lbs) %>%
  filter(date_column >= as.Date("2014-01-01"))


# Create the time series plot for Ratio_Volume_B_A
plot_ratio_volume <- ggplot(Volume, aes(x = date_column, y = Ratio_Volume_B_A)) +
  geom_line() +
  labs(x = "Date", y = "Ratio_Volume_B_A", title = "Time Series Plot of Ratio_Volume_B_A") +
  scale_x_date(date_labels = "%Y-%m", date_breaks = "1 year") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Save the plot
ggsave(filename = file.path(plot_path, "time_series_plot_Ratio_Volume_B_A.png"), plot = plot_ratio_volu
```

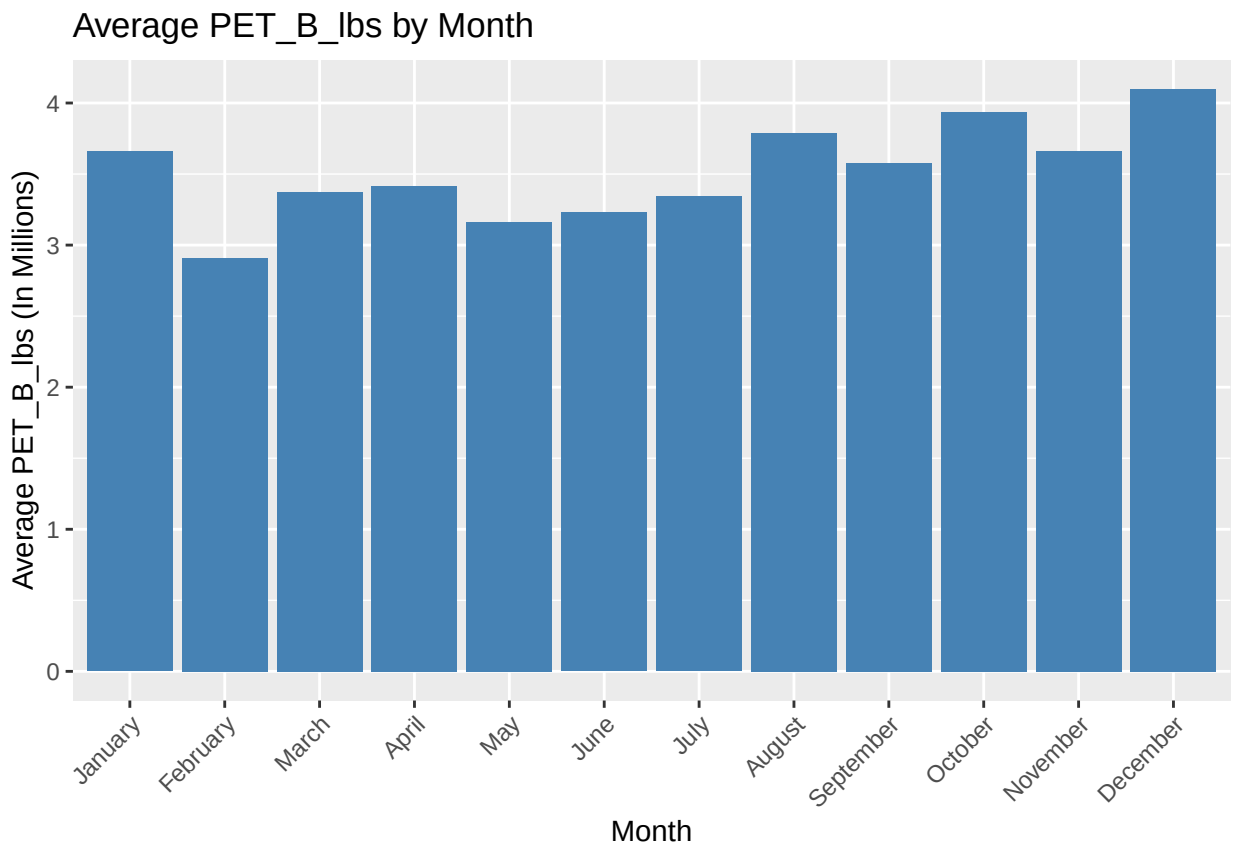
```r
# Optional: print the plot to the console
print(plot_ratio_volume)
```




```r
# Convert PET_A_lbs and PET_B_lbs to numeric
Volume <- Volume %>%
  mutate(PET_A_lbs = as.numeric(PET_A_lbs),
         PET_B_lbs = as.numeric(PET_B_lbs))

# Compute the average PET_A_lbs for each month
average_volume_by_month_PET_A <- Volume %>%
  mutate(Month = format(date_column, "%B")) %>%  # Extract month names
  group_by(Month) %>%
  summarize(Average_PET_A_lbs = mean(PET_A_lbs, na.rm = TRUE)) %>%
  ungroup() %>%
  mutate(Month = factor(Month, levels = month.name))  # Ensure months are in order

# Create the bar graph for PET_A_lbs
bar_plot_PET_A_lbs <- ggplot(average_volume_by_month_PET_A, aes(x = Month, y = Average_PET_A_lbs/1000000
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(x = "Month", y = "Average PET_A_lbs (In Millions)", title = "Average PET_A_lbs by Month") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Display the plot
print(bar_plot_PET_A_lbs)
```

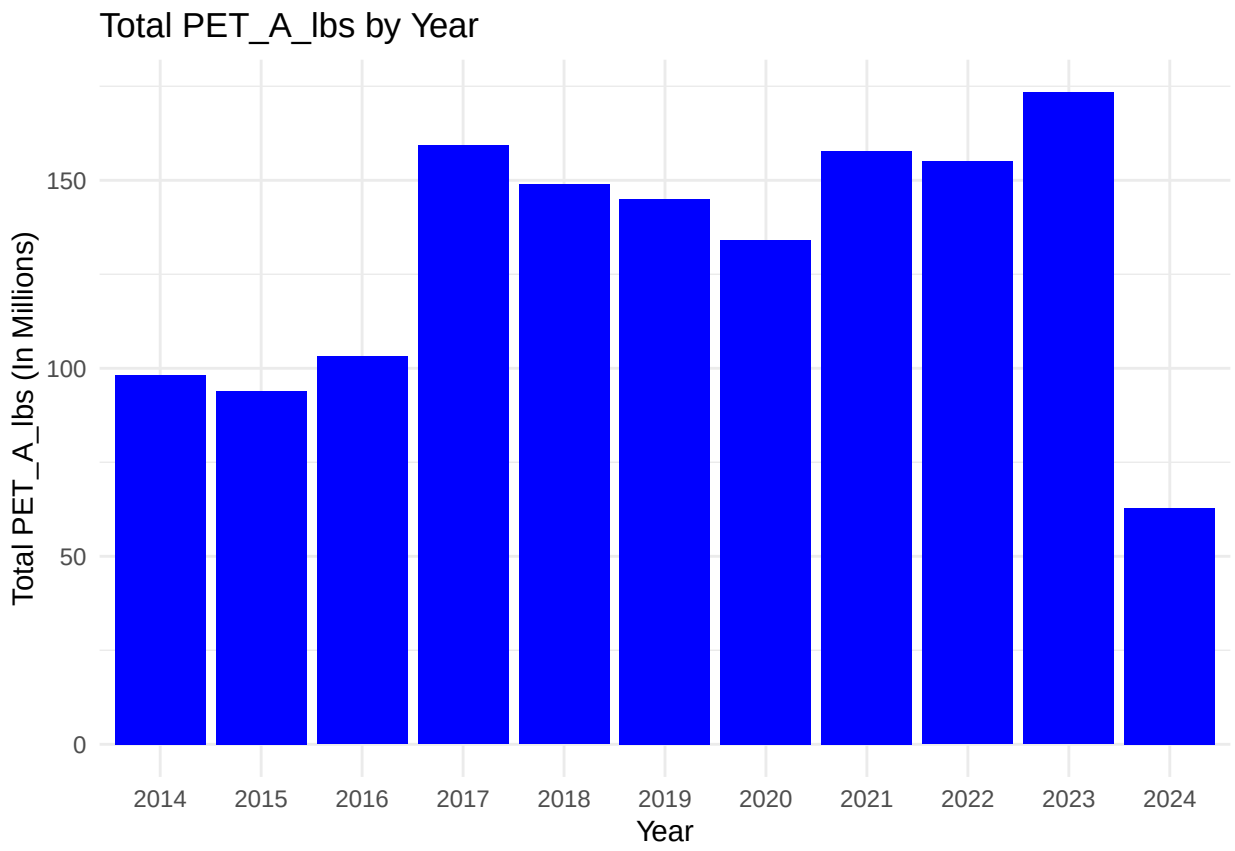



```
# Save the plot
ggsave(filename = file.path(plot_path, "bar_plot_average_monthly_PET_A_lbs.png"), plot = bar_plot_PET_A


# Compute the average PET_B_lbs for each month
average_volume_by_month_PET_B <- Volume %>%
  mutate(Month = format(date_column, "%B")) %>%  # Extract month names
  group_by(Month) %>%
  summarize(Average_PET_B_lbs = mean(PET_B_lbs, na.rm = TRUE)) %>%
  ungroup() %>%
  mutate(Month = factor(Month, levels = month.name))  # Ensure months are in order

# Create the bar graph for PET_B_lbs
bar_plot_PET_B_lbs <- ggplot(average_volume_by_month_PET_B, aes(x = Month, y = Average_PET_B_lbs/1000000
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(x = "Month", y = "Average PET_B_lbs (In Millions)", title = "Average PET_B_lbs by Month") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Display the plot
print(bar_plot_PET_B_lbs)
```



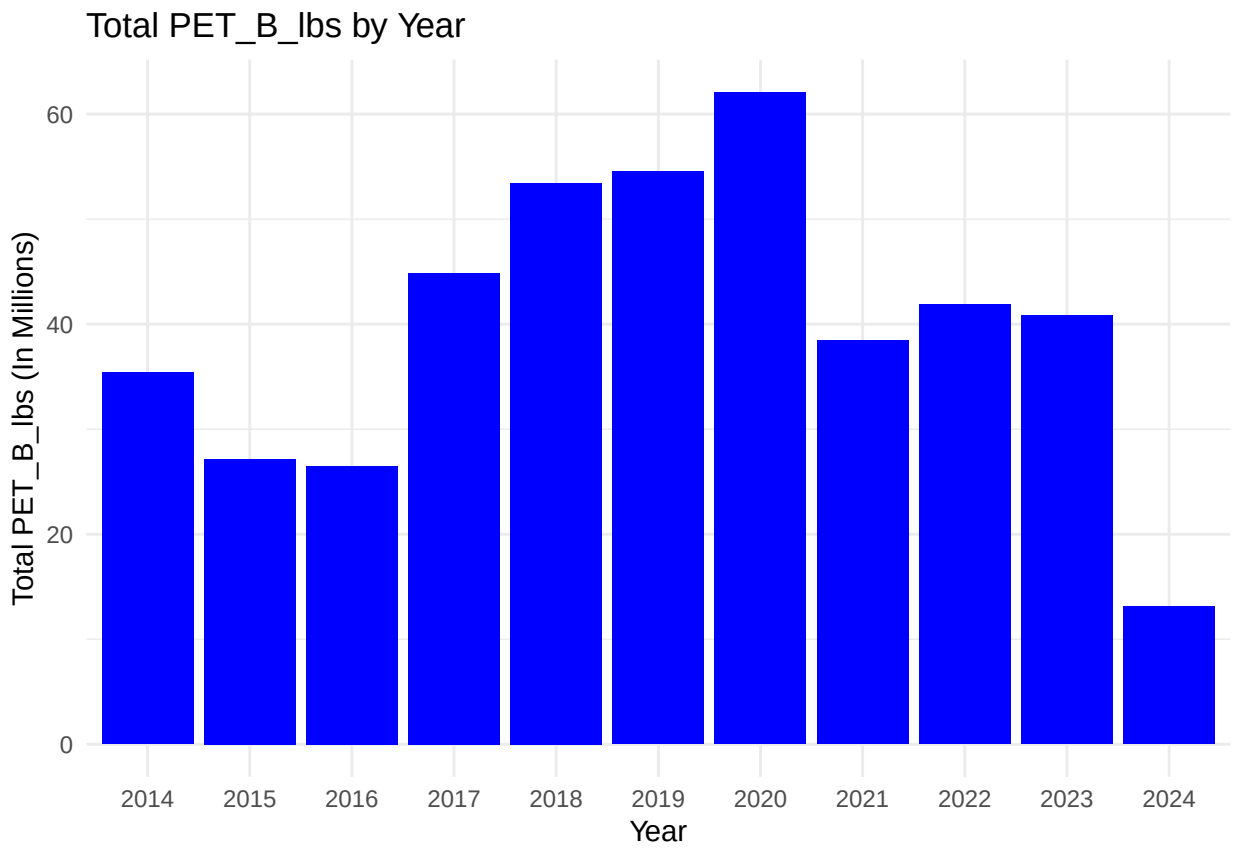

```
# Save the plot
ggsave(filename = file.path(plot_path, "bar_plot_average_monthly_PET_B_lbs.png"), plot = bar_plot_PET_B
```

```
# Group by year and summarize the total for PET_A_lbs and PET_B_lbs
yearly_totals <- Volume_Filtered_after_2014 %>%
  mutate(Year = year(date_column)) %>%
  group_by(Year) %>%
  summarize(
    Total_PET_A_lbs = sum(as.numeric(PET_A_lbs), na.rm = TRUE),
    Total_PET_B_lbs = sum(as.numeric(PET_B_lbs), na.rm = TRUE)
  )



# Create bar graph for Total_PET_A_lbs
bar_plot_PET_A <- ggplot(yearly_totals, aes(x = factor(Year), y = Total_PET_A_lbs/1000000)) +
  geom_bar(stat = "identity", fill = "blue") +
  labs(x = "Year", y = "Total PET_A_lbs (In Millions)", title = "Total PET_A_lbs by Year") +
  theme_minimal()

print(bar_plot_PET_A)
```




```
# Save the PET_A_lbs_by_year plot
ggsave(filename = file.path(plot_path, "bar_plot_Total_PET_A_lbs_by_year.png"), plot = bar_plot_PET_A, w
```

```
# Create bar graph for Total_PET_A_lbs
bar_plot_PET_B <- ggplot(yearly_totals, aes(x = factor(Year), y = Total_PET_B_lbs/1000000)) +
  geom_bar(stat = "identity", fill = "blue") +
  labs(x = "Year", y = "Total PET_B_lbs (In Millions)", title = "Total PET_B_lbs by Year") +
  theme_minimal()

print(bar_plot_PET_B)
```




```
# Save the PET_A_lbs_by_year plot
ggsave(filename = file.path(plot_path, "bar_plot_Total_PET_B_lbs_by_year.png"), plot = bar_plot_PET_B, w
```

```
#run regression for Bid_Pricing

# Filter data to start from July 2013
Bid_Pricing_filtered_B <- Bid_Pricing %>%
  filter(Date >= as.Date("2013-07-01"))

# Function to calculate the number of months since the beginning
calculate_month_number <- function(date_column) {
  # Extract year and month
  year <- as.numeric(format(date_column, "%Y"))
  month <- as.numeric(format(date_column, "%m"))

  # Minimum date you want to consider (adjust as needed)
  min_date <- as.Date("2013-07-01")

  # Calculate difference in months
  diff_in_months <- (year - year(min_date)) * 12 + (month - month(min_date))

  # Add 1 to start counting from 1
  return(diff_in_months + 1)
}

# Add a new column named "no_of_month" to your data
```

```r
Bid_Pricing_filtered_B$no_of_month <- calculate_month_number(Bid_Pricing_filtered_B$Date)

view(Bid_Pricing_filtered_B)

# Fit the linear regression model
model_bid_pricing <- lm(Ratio_B_A ~ no_of_month, data=Bid_Pricing_filtered_B)

summary(model_bid_pricing)
```

```
##
## Call:
## lm(formula = Ratio_B_A ~ no_of_month, data = Bid_Pricing_filtered_B)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.185566 -0.060646 -0.000195  0.041449  0.218627
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.7468952  0.0140559   53.14  < 2e-16 ***
## no_of_month -0.0016114  0.0001848   -8.72 1.19e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07998 on 129 degrees of freedom
## Multiple R-squared:  0.3709, Adjusted R-squared:  0.366
## F-statistic: 76.04 on 1 and 129 DF,  p-value: 1.188e-14
```

```r
export_summs(model_bid_pricing, scale = TRUE, to.file = "xlsx", file.name = "regression_bid_Pricing.xlsx
```

```
## Registered S3 methods overwritten by 'broom':
##   method            from
##   tidy.glht         jtools
##   tidy.summary.glht jtools
```

	Model 1
h	

s predictors are mean-centered and scaled by 1 standard deviation. The outcome variable is in its original units. *** p < 0.001; ** p < 0.01

```r
# Create the scatter plot with regression line
bid_pricing_regression_plot <- ggplot(Bid_Pricing_filtered_B, aes(x = no_of_month, y = Ratio_B_A)) +
  geom_point() +  # Add points
  geom_smooth(method = "lm", col = "blue") +  # Add regression line
  labs(title = "Scatter Plot with Regression Line",
       x = "Number of Months",
       y = "Ratio B/A") +
  theme_minimal()

# Display the plot
print(plot)
```

```
## function (x, y, ...)
## UseMethod("plot")
## <bytecode: 0x000001a5b8f5bda0>
## <environment: namespace:base>
```

```r
# Save the Bid_pricing_regressio plot
ggsave(filename = file.path(plot_path, "bid_pricing_regression.png"), plot = bid_pricing_regression_plo
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```r
#run regression for Volume


# Filter data to start from January 2014
Volume_Filtered_after_2013 <- Volume %>%
  mutate(Ratio_Volume_B_A=(as.numeric(PET_B_lbs)/as.numeric(PET_A_lbs))) %>%
  filter(date_column >= as.Date("2013-07-01"))

# Function to calculate the number of months since the beginning
calculate_month_number <- function(date_column) {
  # Extract year and month
  year <- as.numeric(format(date_column, "%Y"))
  month <- as.numeric(format(date_column, "%m"))

  # Minimum date you want to consider (adjust as needed)
  min_date <- as.Date("2013-01-01")

  # Calculate difference in months
  diff_in_months <- (year - year(min_date)) * 12 + (month - month(min_date))

  # Add 1 to start counting from 1
  return(diff_in_months + 1)
}

# Add a new column named "no_of_month" to your data
Volume_Filtered_after_2013$no_of_month <- calculate_month_number(Volume_Filtered_after_2013$date_column
```

```r
# Fit the linear regression model
model_Volume <- lm(Ratio_Volume_B_A ~ no_of_month, data=Volume_Filtered_after_2013)

summary(model_Volume)
```

```
##
## Call:
## lm(formula = Ratio_Volume_B_A ~ no_of_month, data = Volume_Filtered_after_2013)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.23230 -0.08203 -0.03484  0.06947  0.54264
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.3552925  0.0259651  13.683   <2e-16 ***
## no_of_month -0.0005282  0.0003120  -1.693    0.093 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1259 on 123 degrees of freedom
## Multiple R-squared:  0.02277,    Adjusted R-squared:  0.01483
## F-statistic: 2.866 on 1 and 123 DF,  p-value: 0.09298
```

```r
export_summs(model_Volume, scale = TRUE, to.file = "xlsx", file.name = "regressionVolume.xlsx")
```

Model 1

h

s predictors are mean-centered and scaled by 1 standard deviation. The outcome variable is in its original units. *** $p < 0.001$; ** $p < 0.01$

```r
# Create the scatter plot with regression line
Volume_regression_plot <- ggplot(Volume_Filtered_after_2013, aes(x = no_of_month, y = Ratio_Volume_B_A))
  geom_point() +  # Add points
  geom_smooth(method = "lm", col = "blue") +  # Add regression line
  labs(title = "Scatter Plot with Regression Line",
       x = "Number of Months",
       y = "Ratio_Volume_B_A") +
  theme_minimal()
```

```r
# Display the plot
print(Volume_regression_plot)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```


```r
# Save the Bid_pricing_regressio plot
ggsave(filename = file.path(plot_path, "Volumeregression.png"), plot = Volume_regression_plot, width =
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.